Software Radio-Based Decentralized Dynamic Spectrum Access Networks: A Prototype Design and Enabling Technologies

Feng Ge

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Electrical Engineering

Charles W. Bostian, Chair
Allen B. MacKenzie
Jeffrey H. Reed
Wuchun Feng
Michael S. Hiao

November 30, 2009
Blacksburg, Virginia

Keywords: cognitive radio, wireless communications, software radio, dynamic spectrum access, signal detection and classification, distributed cooperative spectrum sensing

Software Radio-Based Decentralized Dynamic Spectrum Access Networks: A Prototype Design and Enabling Technologies

Feng Ge

(ABSTRACT)

Dynamic spectrum access (DSA) wireless networks focus on using RF spectrum more efficiently and dynamically. Significant progress has been made during the past few years. For example, many measurements of current spectrum utilization are available. Theoretical analyses and computational simulations of DSA networks also abound. In sharp contrast, few network systems, particularly those with a decentralized structure, have been built even at a small scale to investigate the performance, behavior, and dynamics of DSA networks under different scenarios. This dissertation provides the theory, design, and implementation of a software radio-based decentralized DSA network prototype, and its enabling technologies: software radio, signal detection and classification, and distributed cooperative spectrum sensing.

By moving physical layer functions into the software domain, software radio offers an unprecedented level of flexibility in radio development and operation, which can facilitate research and development of cognitive radio (CR) and DSA networks. However, state-of-the-art software radio systems still have serious performance limitations. Therefore, a performance study of software radio is needed before applying it in any development. This dissertation investigates three practical issues governing software radio performance that are critical in DSA network development: RF front end nonlinearity, dynamic computing resource allocation, and execution latency. It provides detailed explanations and quantitative results on SDR performance.

Signal detection is the most popular method used in DSA networks to guarantee non-interference to primary users. Quickly and accurately detecting signals under all possible conditions is challenging. The cyclostationary feature detection method is attractive for detecting primary users because of its ability to distinguish between modulated signals, interference, and noise at a low signal-to-noise ratio (SNR). However, a key issue of cyclostationary signal analysis is the high computational cost. To tackle this challenge, parallel computing is applied to develop a cyclostationary feature based signal detection method. This dissertation presents the method's performance on multiple signal types in noisy and multi-path fading environments.

Distributed cooperative spectrum sensing is widely endorsed to monitor the radio environment so as to guarantee non-interference to incumbent users even at a low SNR and under hostile conditions like shadowing, fading, interference, and multi-path. However, such networks impose strict performance requirements on data latency and reliability. Delayed or faulty data may cause secondary users to interfere with incumbent users because secondary users could not be informed quickly or reliably. To support such network performance, this dissertation presents a set of data process and management schemes in both sensors and data fusion nodes. Further, a distributed cooperative sensor network is built from multiple sensors; together, the network compiles a coherent semantic radio environment map for DSA networks to exploit available frequencies opportunistically.

Finally, this dissertation presents the complete design of a decentralized and asynchronous DSA network across the PHY layer, MAC layer, network layer, and application layer. A ten-node prototype is built based on software radio technologies, signal detection and classification methods, distributed cooperative spectrum sensing systems, dynamic wireless protocols, and a multi-channel allocation algorithm. Systematic experiments are carried out to identify several performance determining factors for decentralized DSA networks.

Contents

List of Figures

List of Tables

Acknowledgements

When I started this dissertation research about three years ago, I even didn't know the exact meaning of modulation. Therefore, it is fair to say that almost all the research in this dissertation started from the knowledge I learned from Charles W. Bostian, Allen B. MacKenzie, Jeff Reed, Wuchun Feng, and Michael S. Hiao, my advisor and committee members. Each has contributed in significant ways to my knowledge repertoire in telecommunications, software defined and cognitive radio, wireless and computer networks, and others, either through my work experiences with them, or in the classroom.

I want to extend a very special thank to Charles W. Bostian. Even though I barely had any knowledge in telecommunications until the end of 2006, Dr. Bostian fully believed in my potential and recruited me as a graduate research assistant just before that year's Christmas. Later, he guided me and also gave me enough freedom to build my knowledge base and explore the world not only in academic research, but also the beyond. His knowledge, open-mindness, and his character have all been exceptional sources from which I have been learning.

I would particularly like to thank Thomas W. Rondeau who introduced GNU Radio to me right after I joined the Center for Wireless Telecommunications (CWT) at Virginia Tech. GNU Radio was extremely helpful for me in learning digital communications, because it is so easy to demonstrate a really working radio system. Tom must have suffered from many of my layman questions before he left Virginia Tech in the summer of 2007.

A lot of thanks go to my colleagues with whom I worked closely. They are Mark D. Silvius, Terry Brisebois, Qinqin Chen, Ying Wang, Alex Young, Rohit Rangnekar, Aravind Radhakrishnan, Sujit Nair, Almohanad Fayez, Bin Le, Paco Garcia, Bin Li, and Mustafa Y. ElNainay. They are all outstanding colleagues. Together we have finished many projects and developed several radio systems and networks within few years.

I also owe gratitude to Judy Hood. She has been always nice to me and helping me go through the administrative and operational processes. In particular, she helped me a lot in improving my written English either directly through herself or through several English student interns she supervised.

Finally, I cannot forget the support from my parents, my girlfriend, my sister, and my brother-in-law. Getting a Ph.D. degree is such a physically and mentally committed journey; they are the people who motivated and supported me to go through it.

Grant Information

This material is based upon work supported by the National Science Foundation under Grant CNS-0519959. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF).

This project is supported by Award No. 2005-IJ-CX-K017 awarded by the National Institute of Justice, Office of Justice Programs, US Department of Justice. The opinions, findings, and conclusions or recommendations expressed in this dissertation are those of the author(s) and do not necessarily reflect the views of the Department of Justice.

This work is also supported by the Defense Advanced Research Projects Agency (DARPA) under grant W31P4Q-07-C-0210. The opinions, findings, and conclusions or recommendations expressed are those of the authors and do not necessarily reflect the views of these sponsors or the official policy or position of the DARPA.

Chapter 1

Introduction

1.1 Background on Dynamic Spectrum Access

When James Clerk Maxwell developed the classical electromagnetic theory which synthesized all previous unrelated observations, experiments and equations of electricity, magnetism and even optics into a consistent theory by his equations - Maxwell's equations – he must have been excited to demonstrate in theory that electricity, magnetism and light are all manifestations of the same phenomenon: the electromagnetic field [1]. However, he might not expect what a foundation he built for the wireless technologies that would be invented later all over the world following his theory. In fact, he even didn't live until Heinrich Hertz became the first person to experimentally verify his prediction that electromagnetic waves exist and propagate with a finite velocity [2]. After that, Marchese Guglielmo Marconi's demonstration of a transatlantic radio link in 1901 began a nonlinear history of wireless technology development that has extended to today and has dramatically changed the world [2].

As of today, wireless technology routinely touches our daily life, from cell-phones to police radios, from TV sets to garage-door openers. Following Maxwell's equations, every wireless device depends on access to the radio spectrum. However, even though wireless technology has advanced dramatically during the past century, radio spectrum in USA has been tightly regulated by the government using primarily one fixed method – *command and control*; by this method, the entire radio spectrum is divided into blocks of frequencies established for a particular type of service over different geographical regions. Most other countries adopt similar policies [3].

Historically, such a policy was made at a time when broadcast radio dominated the applications and a channel would be used almost all the time. However, as radio technologies have advanced and their applications are now diverse, most channels are not used in that

way. When the allocated spectrum is not utilized, and no other applications can access it, it is essentially a waste of spectrum. This fact is verified by many recent spectrum measurements [4, 5, 6]. For example, measurements reported in 2005 by Shared Spectrum Company (SSC) showed that the average spectrum occupancy over all the radio bands between 30 MHz and 3 GHz was 5.2% as observed over multiple typical geographical locations [7].

On one hand, there is only limited *sweet* spectrum, basically from 500 MHz to 5 GHz. This spectrum is desirable because:

- At low carrier frequencies, first there is less absolute bandwidth, and, second, antennas cannot be too small if they are to operate efficiently. For example, at 100 MHz, a quarter-wave antenna would be about 75 cm long, which precludes it from most mobile or portable applications.

- Signals in this range penetrate buildings well and the associated electronic components are inexpensive.

- As frequency increases, signals have poor penetration, and expensive electronic components are needed to support communications. Moreover, at high frequencies, interaction with biological tissues is non-negligible.

This explains why carriers in the late 1990s bid hundreds of billions of dollars of on 3G spectrum [2], and the spectrum utilization efficiency in the cellular bands is at the highest level in reported spectrum measurements [4, 5, 6, 7].

On the other hand, demand in spectrum continuously increases because:

- The increasing use of wireless services, particularly exemplified by the number of mobile phones (It is now more than four billion and has approximately doubled in the past four years) [8].

- The increasing data load being transmitted. For example, according to Cisco Systems Inc., the data load from mobile devices in 2009 is 17 petabytes per month, and it will roughly double in every following year until 2013 [9].

- The emergence of multi-mode products (those including multiple radio functions in a single device).

The limited available spectrum and continuous increase of demand might lead to “spectrum scarcity”, which causes people to think about a possible “spectrum drought” [10]. Such a fear and the sharp contrast of spectrum usage efficiency on different bands (shown

in the above spectrum measurements) led the research community and government agencies to explore a new paradigm in spectrum management – Dynamic Spectrum Access (DSA). DSA allows secondary users opportunistically to use spectrum allocated for privileged users when it is idle while guaranteeing non-interference to those users (or keeping interference below an acceptable threshold). Therefore, DSA's fundamental goal is to best utilize the spectrum resource in all three dimensions: frequency, space, and time.

Further, new mobile services and applications are emerging quickly. However, by the current policy, spectrum re-allocation is difficult and always lags many years behind the need. DSA allows new wireless service providers to access spectrum quickly. Therefore, DSA can bolster wireless technology innovations in the long run.

1.2 Radio Technology Evolution

Supporting DSA requires revolutions in radio and wireless network technologies, wireless standardization and spectrum management policy, and business models [11]. Historically, these domains have co-evolved since the first generation of commercial radios were used. This section intends to briefly review the evolution of radio technology (and its interaction with the regulatory policies), and to argue that now is the time to realize DSA, at least from the technology perspective.

Modern wireless systems are the result of advances in information and control theory, signal processing, electromagnetic field theory, circuit design, and computing technology. But historically, the early stage (before 50's) of radio technology revolution was primarily focusing on interference control, communication performance, and cost and power consumption reduction. For example, the first generation of Morse-code-based spark-gap transmitters (a very broadband device) immediately had dysfunctional interference problems, when other people tried to duplicate Marconi's success and deployed similar radios in 1901 [2]. This first "spectrum scarcity" led to invention of carrier based radios. Before Edwin H. Armstrong invented FM radios in 1933, research in radio technology was focused on AM radio and its architecture optimization. Prior to the discovery of the ionosphere, people thought that the only high frequency (HF) propagation mode was diffraction around the curvature of the earth. The lower the frequency, the less was the diffraction loss. Everybody wanted to get as close to DC as they could. Therefore, a temporary "spectrum scarcity" happened in the 20's. The result was that radio amateurs were forced by the regulatory agency to use frequencies above 1.5 MHz. However, the amateurs quickly showed that 1.5-30 MHz is quite useful for long-distance communication via the ionosphere (thanks to the discovery of the ionosphere's existence by several scientists [12]), therefore, resolv-

ing this "spectrum scarcity" problem. After that, the usable carrier frequency in radio technologies was continuously pushed higher and higher for many years.

Meanwhile, information theory was developed, culminating in 1948 with the famous *channel capacity theorem* developed by Claude Shannon. This provides a theoretical foundation establishing the relation between information rate and bandwidth in the presence of noise. Shannon's theorem is of significant importance to illustrate the "spectrum scarcity" problem. Here a brief introduction to the theorem is needed. For the special case of a band-limited channel corrupted by stationary additive white Gaussian noise (AWGN), the *channel capacity* C, in bits per second, is determined as:

$$C = B\log_2(SNR + 1) \tag{1.1}$$

where B is the channel bandwidth (in hertz) and SNR is the signal-to-noise ratio over that bandwidth. Shannon's theorem contains much more than this, but Eq. 1.1 is enough to illustrate the point. When SNR is fixed (so is the signal transmitting power), according to Eq. 1.1, there is a theoretical limit on the data rate that can be transmitted via a radio link, and this limit increases linearly with the available bandwidth. However, when the bandwidth is fixed, the signal transmitting power must increase exponentially to support a linear increase in the data rate. Therefore, from a resource usage perspective, increasing the channel bandwidth is the more efficient way to support a higher data rate. However, if the total available bandwidth is limited (this is true in some sense, at least for *sweet* spectrum, as we saw above), the total aggregated data rate is limited (so is the number of users). On the other hand, to raise the data rate over a fixed bandwidth, an exponential increase in power is needed. Accordingly the signal's interference region increases exponentially, thereby reducing the number of users that can co-exist in a geographic region. This is related to an important concept, *frequency re-use* applied by cellular systems.

The next thirty years saw the prosperity of transistor, integrated circuit (IC), and computing technology. Meanwhile, the cellular concept emerged in late 40's to solve another "spectrum scarcity" – an over demand in mobile telephone service operating in 150-MHz band since 1946 [2]. However, it went through decades of technology development and then years of hearings before the trial service was finally deployed in 1978 in Chicago, Illinois [2]. The first generation cellular system in the USA, Advanced Mobile Phone System (AMPS), was an analog FM-based system, using frequency-division multiple access (FDMA) and supporting frequency-division duplexing (FDD). The combination of frequency re-use and hand-off in cellular systems dramatically increased the available number of channels in many geographic regions, therefore solving the over demand problem,

and started a novel and astonishingly successful business model.

It is relevant that cellular systems were finally made practical by advances in IC and computing technology. The network management involved in implementing frequency re-use and hand-off requires a significant computational load. Without modern IC and computing technology, the hardware would be impractically cumbersome. This is even more true for code division multiple access (CDMA) based second generation cellular systems. Mitigating the near-far problem during hand-off in such systems requires tremendous amounts of computation, and CDMA handsets are consequently complex in terms of signal processing [2].

With the continuous cost reduction in IC manufacture, digital radios became inexpensive and began to dominate because of their clear advantages [13]:

- A digital receiver need only distinguish between a finite number of waveforms, therefore it is possible to recover digital information distorted by noise and interference.
- Many signal processing techniques are available to improve system performance: source coding, channel coding, equalization, encryption.
- Digital ICs are inexpensive to manufacture and even complex chips can be mass produced at low cost.
- Digital radios can integrate voice, video, and data into a single system.
- Digital communications systems provide a more flexible tradeoff between bandwidth efficiency and energy efficiency than analog communications.

One good example to show digital radio's advantages is Global Positioning System (GPS), which suffers from large path loss and Doppler frequency shift. Based on highly sophisticated application-specific integrated circuit (ASIC), a modern GPS device has a tiny size and is just one sub-function of most smart-phones. Further, digital radios are able to support a high data rate by exploiting a wider frequency bandwidth, providing complex waveforms and performing coding and error correction. This is exemplified by the data rate increase in 3G and 4G cellular systems. The culmination is multiple-input and multiple-output (MIMO) based CDMA or orthogonal frequency-division multiple access (OFDMA) radios containing sophisticated ASICs.

The above trend is a continuous optimization of spectrum utilization efficiency and radio resource usage. As the digital proportion of radios and the computing ability (following Moore's Law) continuously increase, functions realized previously in highly optimized ASICs at the hardware level can now be supported in the software domain. Coined by

Mitola in 1992 [14], *software (defined) radio* (SDR) becomes feasible. By the IEEE definition, SDR is a type of radio in which some or all of the physical layer functions are software defined [15]. It promises a high level of reconfigurability and flexibility, and has a lot of advantages, for example [16]:

- multi-mode capability,
- reduced content of expensive custom silicon,
- reduced parts inventory,
- DSP can compensate for using cheaper RF components,
- lower life-cycle cost,
- faster time to market.

Quite different from previous radios whose functions are predetermined and hardware based, SDR further allows radios to be adaptive. Communication systems can monitor their own performance, and reconfigure their own parameters to improve their performance by some form of closed-loop action.

Even further, such adaption can be intelligent and radios can even learn from different contexts. Mitola envisioned such a radio – *cognitive radio* in his dissertation [17]. Cognitive radio (CR) is aware of its environment and internal state, and can make decisions about its operating behavior based on that information and predefined objectives [15]. During the past 10 years, research in CR has proliferated [18, 19]. In particular, machine learning and artificial intelligence are now widely applied in CR research [20, 21, 22].

As implied above, radio technology and spectrum management policies have co-evolved. Historically, policies have dominated the relationship. Unfortunately new polices almost always lag behind new technologies, examples include the enactment of policies for the AMPS system (11 years later), Personal Communications Service (PCS) (6 years later), and Advanced Wireless Services (AWS-1) (6 years later) [23]. However, often when a novel technology successfully forced the enactment of a new policy, this technology would later bring significant social and economic benefits. The most prominent case is the cellular technology. As of today, it is the largest wireless communications market and there are more than four billion mobile phone users worldwide [8].

However, there was a remarkable exception to the usual interaction between radio technology revolution and spectrum management policy evolution. The Federal Communications Commission (FCC) began to consider the now-famous unlicensed band operation in

1981 [24]. It released in 1985 the "First Report and Order (FRO)" which consists of the original set of rules governing unlicensed access to the industrial, scientific and medical (ISM) bands. The following 25 years have witnessed a steady growth in the unlicensed device industry such as cordless phones, sensor devices, remote controls, and outdoor point-to-point links [24]. In particular, the wireless local area network (WLAN) is today a ubiquitous unlicensed device in our daily life. The Wi-Fi market has become the second largest wireless communications market in the world (behind only cellular telephony) and for 2010 is projected to approach 1 billion units. Therefore it is no surprise to see that spectrum utilization efficiency of the ISM band is among the highest levels indicated by recent spectrum measurements [4, 5, 6, 7].

In a broad sense, dynamic spectrum access has already been employed in existing wireless networks such as cellular networks, cordless phone systems, and WLANs. But DSA only happens on limited bandwidth in those networks. Such networks' wide deployment and high spectrum utilization efficiency are encouraging the application of DSA to even broader bands. Today's technology in radio systems and networks as well as computing devices is mature enough to make this happen. For example, the first trial deployment of a white spaces network announced in October, 2009 in Virginia [25] brought (or will bring) broadband Internet to rural America over unused TV broadcast airwaves. Furthermore, applying software and cognitive radio in DSA applications will revolutionize communications [18].

1.3 Current DSA Technology Efforts

Dynamic spectrum access can be applied in centralized networks like cellular systems, or decentralized networks like mobile ad-hoc networks (MANETs). This section briefly review some current research efforts in the DSA technology. It gives a short introduction to four IEEE standards involving DSA, and then presents a broad survey of research activities in decentralized DSA networks.

1.3.1 Centralized DSA Networks

Applying DSA in centralized networks has made substantial progress during the past ten years. It is represented by four wireless standards: IEEE 802.11h, IEEE 802.11y, IEEE 802.16h, and IEEE 802.22. This section briefly introduces their DSA working mechanism.

IEEE 802.11h

IEEE 802.11h (or IEEE 802.11h-2003) is an amendment added to the IEEE 802.11 standard for providing Dynamic Frequency Selection (DFS) and Transmit Power Control (TPC) to the IEEE 802.11a MAC. According to the amendment, WLANs share spectrum with primary-use devices like satellites and military radar systems in the 5-GHz frequency band [26].

DFS enables WLAN devices to detect other devices using the same radio channel; once detected, the devices operate on another channel if necessary. DFS is responsible for avoiding interference with other devices, such as radio systems and other WLAN segments, and for uniform utilization of channels. *An access point specifies the usage of DFS in the data frames that WLAN stations use to find the access point*. The access point initiates a channel switch by sending a frame to all stations associated with the access point.

TPC is intended to reduce interference from WLANs to satellite services by reducing the radio transmitting power WLAN devices use. TPC can also be used to manage the power consumption of wireless devices and control the range between access points and wireless devices. *An access point specifies TCP support in the frames it transmits to WLAN stations*. Radio power in a WLAN segment can be adjusted to reduce interference with other devices while still maintaining sufficient link margin for data communication in the wireless network.

IEEE 802.11y

IEEE 802.11y is an amendment to IEEE 802.11-2007 standard that will enable high powered Wi-Fi equipment to operate on a secondary basis in the licensed frequency band (3650 to 3700 MHz) in the United States, except when near a grandfathered satellite earth station [27]. The above equipment can operate at distances of 5 kilometers or more supporting applications like back hauls for municipal Wi-Fi networks and public safety and security networks.

According to the FCC's final rules for a novel "lite licensing" scheme in the 3650-3700 MHz band [27], licensees only need to pay a small fee for a nation wide, non-exclusive license. However, they will pay an additional nominal fee for each high powered base station that they deploy. Licensing is not required by either the client devices (fixed or mobile) or their operators. *However, these devices must receive an enabling signal from a licensed base station before transmitting. All stations must be identifiable in the event they cause interference to incumbent operators in the band.* According to IEEE 802.11y, multiple licensees' devices use a *contention based protocol* when transmitting in the same area.

Licensees are required to resolve the dispute between themselves if interference between them, or the devices that they have enabled, cannot be mediated by technical means.

IEEE 802.16h

IEEE 802.16h specifies improved policies and medium access control enhancements, to enable coexistence among license-exempt systems based on IEEE Standard 802.16 (or WiMAX) and to facilitate the coexistence of such systems with primary (non-WiMAX) users [28]. It targets at local and metropolitan area networks in the 5.8 GHz, 3.65 GHz, and other frequency bands.

This standard specifies two architectures for networks of base stations (BSs) [28].

- Ad-hoc network architecture. BSs use cognitive radio signaling to interact with each other following a set of coexistence protocols which regulate BSs to share data among them and to provide service to subscriber stations (SSs). There is no authentication server.

- Centralized network architecture. A distributed local BS identifier servers (BSISs) and Radius are deployed with a centralized Root Radius Server. Each service BS needs authentication by the local Radius server. BSs learn its neighbors by querying BSIS of different network and coexistence time slot mechanism.

IEEE 802.22 Wireless Regional Area Network (WRAN)

IEEE 802.22 is a standard for Wireless Regional Area Networks (WRAN) using white spaces in the TV frequency spectrum between 54 and 862 MHz [29]. The development of the IEEE 802.22 WRAN standard is aimed at using cognitive radio techniques to allow sharing of geographically unused spectrum allocated to the Television Broadcast Service on a non-interfering basis. It mainly aims at extending broadband access in low population density rural areas by using the TV broadcast bands for their better signal propagation characteristics.

The IEEE 802.22 WRAN will be a centralized system consisting of base stations (BSs) and customer-premises equipment (CPE). All the BSs are connected to Authentication, Authorization and Account Servers through an IP network. Each BS and CPE is capable of sensing. The CPEs will be sensing the spectrum and will be sending periodic reports to the BS informing it about what they sense.

Upon initialization, each BS has to consult TV usage database and regional WRAN information database to identify candidate channels, then perform sensing to confirm vacancy of channels, and finally establish operation on a vacant channel. The CPE will scan

previous list of candidate channels or all downstream channels until it finds a valid downstream signal. After acquiring super-frame control header (SCH) which contains information about system type, channel, channel bonding, quiet periods, the CPE will sense on all relevant channels surrounding the operating channel, and obtain upstream and downstream parameters and perform initial ranging.

1.3.2 Decentralized DSA Networks

The above four technologies are primarily focused on centralized wireless networks, which are well studied and standardized. However, supporting DSA in decentralized wireless networks drives the complexity both within individual nodes and over a whole network to an unparalleled level [30]. Both innovative protocols and novel architecture are needed. For example, the most basic requirement in DSA networks is to guarantee non-interference to incumbent users while using RF spectrum more efficiently and dynamically. This requires a new set of functions related to spectrum sensing at the PHY layer. Node cooperation protocols and possible policy modules are also needed [30]. Compared to conventional MANETs, the coupling of channel allocation, power control, and topology control is more likely in DSA networks because of the dynamics in the radio environment and over the network [31, 32, 33]. Further, cross-layer architecture is fundamentally necessary in DSA wireless networks because the operating frequency band depends on the channel occupancy measured at the PHY layer and directly impacts the above layers both within and across nodes [34]. Moreover, new protocols and methods like Disruption Tolerant Networking (DTN) protocols and Content Based Access (CBA) techniques are also being included in DSA network design to cope with network connection disruptions under different environments [35, 36].

Partly because of the above system complexity and partly because of the unavailability of a suitable open platform [37], few physical networks have been built, even at a small scale, to investigate the performance and dynamics of decentralized DSA networks. Nonetheless, good theoretical analyses and simulations have been carried out. For example, coupled power control and channel allocation optimization are heavily investigated [32, 33]. Game theory is introduced to propose new protocols in spectrum sharing and to investigate individual nodes' behavior in cognitive radio networks [38, 39].

Widely known in this research community are two built networks: DARPA's XG [30] and Wireless Network after Next (WNaN) projects [36]. The first generated several publications which present significant results and insights in developing spectrum sensing methods, building individual DSA nodes, and testing overall systems in different network sce-

narios and RF environments [30, 31]. For example, it requires a complicated software system in XG radios to support four principal parts: a DSA engine, a sensing subsystem, a communication subsystem, and a policy module [30]. Since several functions must run simultaneously, multi-threading is essential to the system design. Published results describing field tests report that a sensing delay at even one node can dramatically degrade the DSA network performance and cause interference to incumbent users [31]. Overall, this project reflects the complexity of DSA node architecture and some unique dynamics in DSA networks under different RF environments. However, the published results from the XG project are based on a six node network [30, 31]. An important goal of WNaN is to scale the network size up to about a thousand nodes [36]. This scaling requirement, together with newly included functions and algorithms like CBA and DTN [36] will dramatically increase the system complexity. Unfortunately, few details about the project are yet publicly available as the present paper is written.

As intelligence and "cognitive" capabilities are increasingly introduced into wireless nodes and network protocols, cognitive radios [20, 22] and cognitive radio networks [40, 18] can support better communication quality of service (QoS) while more efficiently using resources like RF spectrum, network bandwidth, and battery power within each node and over a whole network. Nonetheless, supporting cognitive capabilities in DSA nodes and networks will require "cognitive" or "learning" algorithms and protocols, which undoubtedly will further increase the node and network complexity [40]. Further, the performance of cognitive radio networks, particularly large scale ones with a decentralized structure, will likely have dynamics that are not yet explored [41].

1.4 The Dissertation

Considering the complexity of both the individual nodes and the network in decentralized DSA networks, experiments are necessary to gain insights into network design and to investigate network performance under different scenarios [31]. For example, a network protocol for power control must consider newly needed functions, their computing and communication cost, and the associated increase of system/network complexity. Otherwise, achievable power performance derived from theoretical analysis or simulation will not be approached [33]. Further, in research of complex systems like cognitive radio networks, experimental methods can access system/network conditions that may be neglected by either theoretical analysis or computational simulation.

The author's view of decentralized DSA networks is shown in Figure 1.1. In this network, nodes opportunistically use vacant channels when incumbent users are not transmit-

ting. The whole network consists of decentralized mobile nodes and works in a dynamic radio environment where fading and interference exist and incumbent users may appear at any time. Because possibly hostile radio environments may introduce shadowing and fading, distributed cooperative spectrum sensing is necessary [42]. Each node can perform functions like signal detection and classification, node reconfiguration, and some network protocols. However, it is power inefficient to enable all those functions on all nodes simultaneously [43]. For example, if a cooperative sensing network is properly partitioned into clusters [43], as shown in Figure 1.1, only a few nodes are needed within each cluster for signal detection and classification. But one node must work on data fusion in each cluster, and there must be some network level sensing data exchange and synchronization. To increase network battery life and reduce network communication overhead, the network dynamically selects nodes to assume functions like sensing and data fusion.

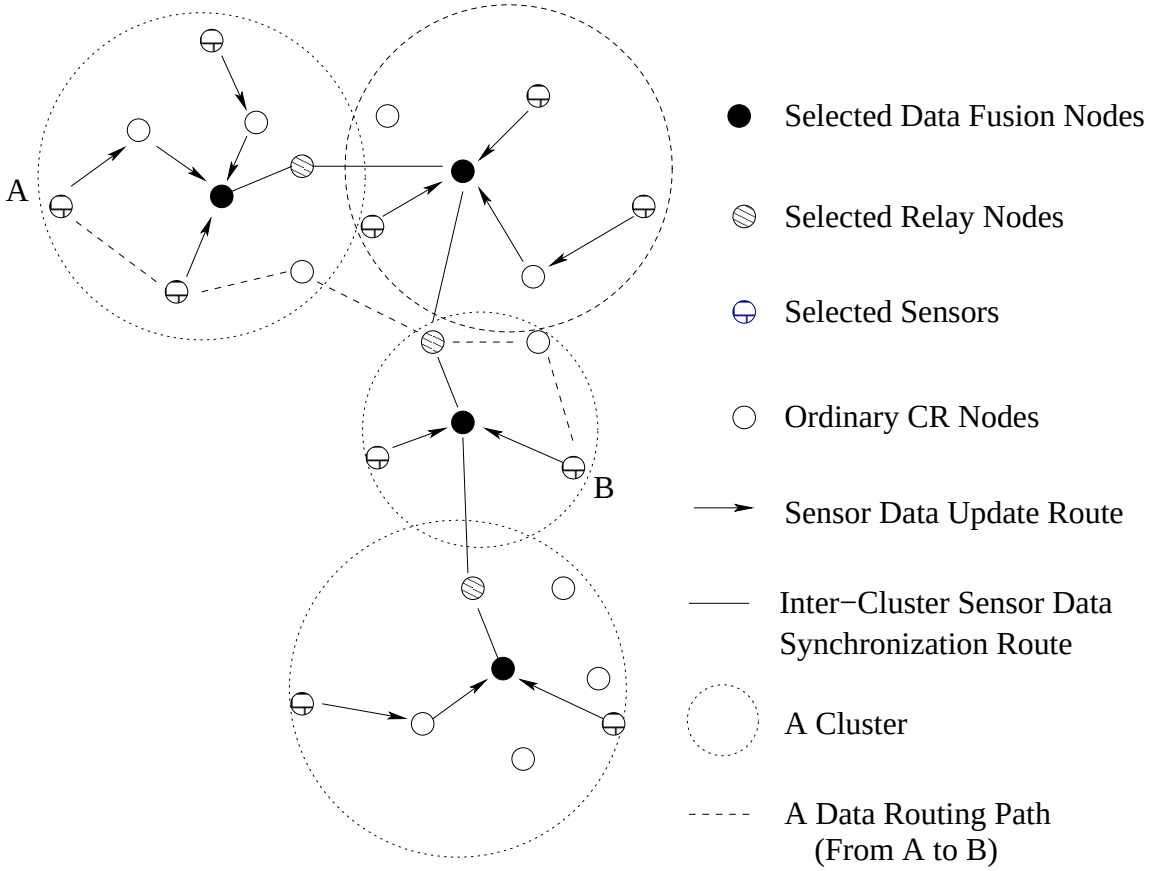


Figure 1.1: The author's view of a decentralized DSA network.

This dissertation, by taking a bottom-up approach, provides the theory, design, and implementation of a decentralized CR and DSA network prototype corresponding to Figure 1.1, which enables distributed cooperative spectrum sensing, multi-channel allocation,

and dynamic spectrum access. Research and development of this network prototype intends not only to provide a system/network level design, but also through systematic experiments to identify performance determining factors in decentralized DSA networks.

To achieve such goals, research is needed not only at the system level, but also for individual underlying technologies. Therefore, this dissertation starts first in Chapter 2 with an investigation of performance issues of SDR [16], a critical enabling technology in the dissertation research specially, and for CR and CR networks in general. Chapter 3 reviews different methods in signal detection and classification, the most popular approach used in DSA networks to guarantee noninterference to primary users. It then focuses on developing a parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading [44]. The work described in Chapter 2 and 3 provided important background information for the network experimental work in Chapter 5. The investigation of SDR performance limitations was important for setting the power budgets and assigning computational tasks to the radio nodes. The work on parallel computational approaches and cyclostationary techniques in spectrum sensing was in one sense a dead end, since the available experimental nodes were not powerful enough to implement these methods. But the author thinks they will be implemented in subsequent networks based on more powerful platforms (the DARPA WNaN radio for example). For that reason, they are presented here.

After a brief review of research progress in decentralized detection from a hypothesis test perspective, Chapter 4 explores performance issues in cooperative spectrum sensing, a necessary system in DSA networks which need a complete semantic map of the radio environment at any time [42, 45]. The author provides a set of schemes for spectrum data flow and process management within spectrum sensors and DSA brokers, which are used in a cooperative spectrum sensor network requiring low data latency and high data reliability. This chapter also introduces a cooperative spectrum sensing system and its component signal sensors (developed by the author and his colleagues). Based on the above enabling technologies, Chapter 5 describes the design and implementation of a decentralized and asynchronous DSA network prototype, and its constituting functions and components. It then presents the performance results of this network prototype in cooperative spectrum sensing, multi-channel allocation, and dynamic spectrum access. Chapter 6 concludes the dissertation and looks forward to some future research.

Throughout this dissertation, terms will be used in consistence with IEEE Standard Definitions and Concepts for Dynamic Spectrum Access: Terminology Relating to Emerging Wireless Networks, System Functionality, and Spectrum Management, IEEE Std 1900.1-2008^{TM} [15] unless otherwise noted.

1.4.1 Contributions

Overall, this dissertation research makes contributions to the following subjects:

1. It builds a distributed cooperative spectrum sensing network with a set of novel schemes for spectrum data flow and process management within its spectrum sensors and DSA brokers. This network provides DSA networks a complete semantic map of the radio environment while achieving low data latency and high data reliability. Further, each DSA broker, based on results from its associated sensors, provides spectrum usage statistics of sensed channels.

2. It designs and implements a software radio based decentralized DSA network prototype which enables distributed cooperative spectrum sensing, multi-channel allocation, and dynamic spectrum access. Through systematic experiments, it identifies several performance determining factors for decentralized DSA networks.

3. It addresses three practical issues governing SDR performance that are critical in CR and DSA network development: RF front end nonlinearity, dynamic computing resource allocation, and execution latency. Detailed explanations and quantitative results are provided.

4. It presents a parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading.

Chapter 2

Software Defined Radio Performance Investigation

Software Defined Radio (SDR) technology has been widely embraced as the way to develop waveform and frequency agile radio platforms. It is a promising technology for cognitive radio and DSA network development. However, today's SDR systems, particularly inexpensive ones which are widely used for CR and DSA network prototype development (including this dissertation), tradeoff their radio performance to achieve high flexibility and reconfigurability.

Several major technologies in this dissertation research rely heavily on SDR, including radio signal detection and classification as well as a reconfigurable PHY layer and a MAC layer. Therefore, a performance study of SDR is needed before using it in those functions. This chapter addresses three practical issues governing SDR performance that are critical in DSA network development: RF front end nonlinearity, dynamic computing resource allocation, and execution latency. It provides detailed explanations and quantitative results of how well SDR performs. This work, while it may seem to the reader to be a digression from the main themes of this dissertation, was essential to the design of both the individual nodes and the overall prototype DSA network.

2.1 SDR Overview

Aided by advances in silicon technology, radio frequency (RF) technology, and software methods, SDR engineers have moved digital signal processing functions progressively closer to the radio antenna [14]. SDRs can now replace a lot of functionally inefficient analog circuitry with reliable, low-priced digital circuits. Furthermore, the flexibility of

programmable signal processors augments an SDR's ability to accommodate various waveform formats and protocols on one radio platform, and allows the SDR to configure itself "on the fly"; today, SDRs can achieve true multi-band multi-mode reconfigurability [16]. SDR technology has been widely embraced as the way to offer functionalities for CR and DSA developments.

However, currently it is challenging to create an RF front-end and associated AD/DA converters that are applicable to a variety of signals with widely differing center frequencies, modulation bandwidth, and power levels [16]. Therefore, SDR's current performance still depends heavily on analog RF technologies. For example, RF front-end performance metrics, particularly non-linearity and dynamic range significantly limit SDR's.

From a user perspective, an easy development environment is needed for SDR systems. However, this is challenging for digital signal processors (DSP) or field-programmable gate arrays (FPGA). Fortunately, general purpose processors (GPPs), which have been continuously increasing their computational ability following Moore's Law, are now fast enough to do a lot of real time digital signal processing tasks and functions. With many library functions and an easy development environment, several GPP based SDR architectures like GNU Radio [46] and OSSIE [47] are developed and widely used in the SDR research community. However, this makes SDR not just a radio or digital signal processing problem anymore. It also becomes a computer problem; GPP's architecture [48] and operating system (OS) mechanism [49] should be considered. Further, the interface and control of a SDR RF front end by the GPP should also be considered.

The GPP based open source SDR development toolkit GNU Radio [46] and its typically associated RF front end, Universal Software Radio Peripheral (USRP) [50] are critical components in this dissertation's research. However, performance limitations exist. This chapter addresses three practical issues governing a SDR system comprised of GNU Radio and a USRP: RF front end nonlinearity, dynamic computing resource allocation, and execution latency. Quantitative investigation results of such a SDR system not only guide this dissertation's network prototype design and development, but also serve as a foundation for identifying those performance determining parameters in general DSA networks.

2.2 GNU Radio and USRP

USRP is an openly designed low-price SDR hardware platform which implements radio front-end functionality and A/D and D/A conversion, connected to the PC that hosts the device. We discuss USRP performance in detail here because it is by far the most widely used platform in academic SDR research and the one from which our prototype network

was built.

Two generations of USRPs have been made; more details are available in [50]. USRP I, used in this dissertation research, consists of a motherboard with four high speed 12-bit 64 Msps analog to digital converters (ADC), four high speed 14-bit 128 Msps digital to analog converters (DAC), an Altera FPGA and a programmable Cypress FX2 USB 2.0 controller. The ADCs, DACs and the FPGA together provide support for IF processing. The FPGA on the board provides four digital up converters (DUC) and four digital down converters (DDC) to shift frequencies between baseband and the required operating frequency. The FPGA can be reprogrammed to provide additional functionality like pulse shaping. RF front ends are attached in the form of daughter cards which can currently cover all the radio bands from 0 Hz to 2.4 GHz. A new wide-band daughter-board based on the Motorola Radio Frequency Integrated Circuits (RFIC) technology is under development. More details are available in [51].

GNU Radio is an open source toolkit for building software radios [46]. It was started in early 2000 by Eric Blossom and others and has evolved into a mature software infrastructure used and supported by a large community of developers. It was originally designed to run on General Purpose Processors (GPP), combined with minimal analog radio hardware, and allows software radio development of waveforms, modulations, protocols, signal processing, and other communications functions in the digital domain. The GNU Radio signal processing library includes existing and developing blocks for most signal processing functions, such as waveform modulation and filter creation. It also includes I/O operations like file access. Programming in the GNU Radio platform uses a combination of C++ and Python, a simple, high-level language: the computationally intensive processing blocks are implemented in C++ while the control and coordination of these blocks for applications that sit on top are developed in Python. The USRP is fully supported by the GNU Radio library and a combined system of both is shown in Figure 2.1.

2.3 RF Front End Nonlinearity

Current SDR performance still depends heavily on analog radio frequency (RF) technologies. Inter-modulation and other nonlinear effects in these devices make it challenging to create an RF front-end that is applicable to a variety of signals with widely differing center frequencies, modulation bandwidths, and power levels. Unanticipated inter-modulation products can seriously degrade receiver performance. For example, nonlinearity not only impacts radio performance and power consumption [52], but also fundamentally limits CR's application in dynamic spectrum access (DSA) [53] and network resource optimiza-

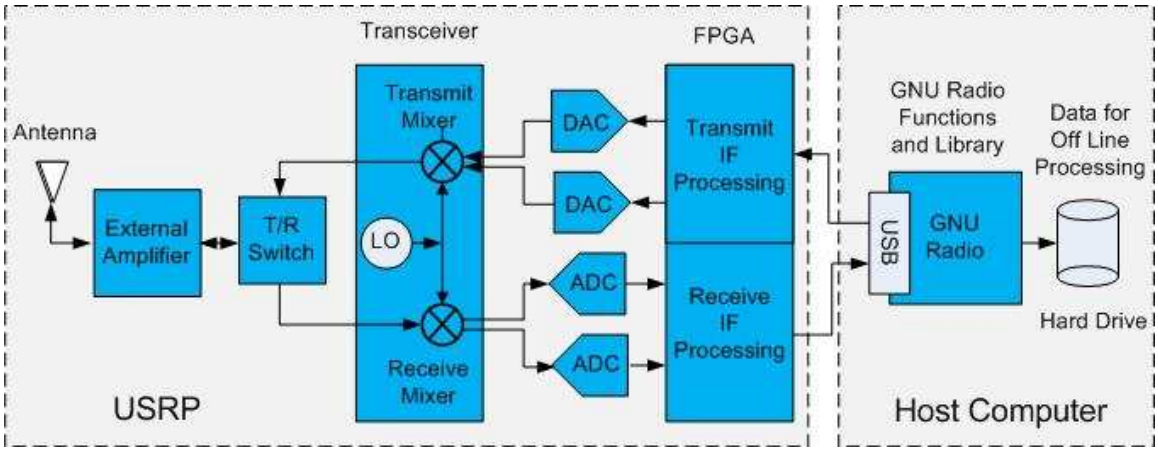


Figure 2.1: A basic SDR system based on GNU Radio and USRP. ©2009 IEEE. Reprinted, with permission, from Ge et al., "Cognitive radio: From spectrum sharing to adaptive learning and reconfiguration," IEEE Aerospace Conference, Big Sky, MT, 2008.

tion [54]. Those problems exist so widely that Marshall proposes to use some CR mechanisms to mitigate them and gain performance benefits in radio operation reliability, cost and reduced energy consumption [52].

This section's goal is to present some analysis on the impact of RF nonlinearity on SDR performance through USRP and GNU Radio [46]. It gives some quantitative results to identify a right set of parameters for using USRPs in DSA network development. These results guided our development of the prototype DSA network.

Fundamentally, nonlinearity comes from nonlinear elements like diodes and transistors that are used in RF amplifiers and mixers. If an input signal, for example, a sinusoidal signal $A = \cos(2\pi f_0 t)$ has a power level that lies beyond the linear region of such devices, the output signal will include frequency components at f_0 as well as at $2f_0$, $3f_0$, etc. Further, if there exists more than one input signal with one or more having power levels beyond the linear regions of such devices, they will interact with each other and degrade the overall performance on each signal. There are mainly three nonlinearity problems: inter-modulation distortion, desensitization, and cross-modulation [16].

2.3.1 Types of Non-linearity Distortion

Inter-modulation

The RF front end of a USRP is shown in Figure 2.2, where GNU Radio controls the receiver gain through the programmable gain amplifier (PGA) before the ADC. To illustrate the inter-modulation, the author sets up two transmitters with the same power levels and at

close center frequencies (900MHz and 900.2MHz in Figure 2.3). Both transmits Gaussian minimum-shift keying (GMSK) modulated data with a baseband data rate of 50kB/s. A receiver captures 4MHz bandwidth of signal centered at 900MHz. When the receiving gain in the PGA is set at 40% of the maximum value, only two signals are shown in the frequency domain. However, as the receiving gain is increased to 45%, multiple signals appear, as shown in Figure 2.4. The additional signals are third and fifth-order inter-modulation products that appear in the IF passband and cause downstream interference. At the receiver output they are indistinguishable from unwanted signals that exist at the input.

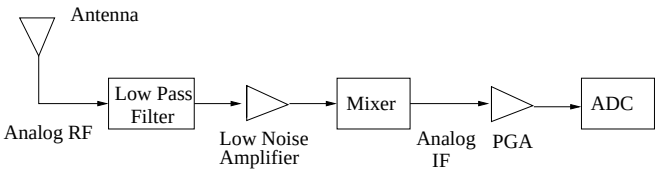


Figure 2.2: The RF front end in the USRP. ©2009 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "SDR implementation issues: RF front end nonlinearity and dynamic computing resource allocation," Software Defined Radio Forum, Washington DC, 2009.

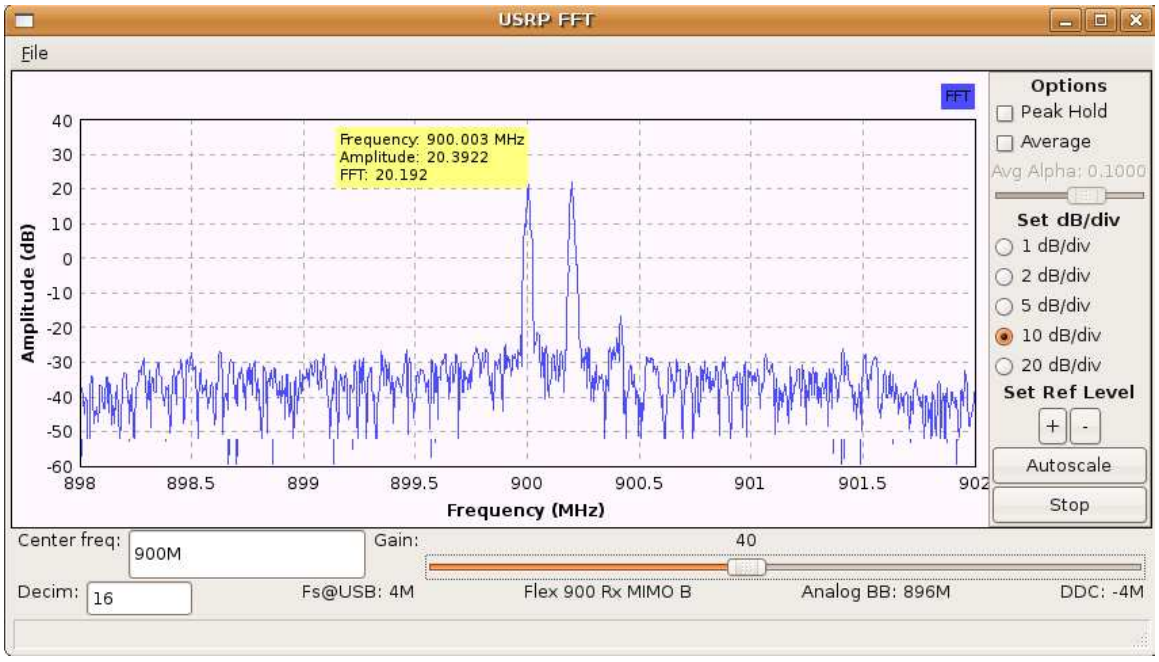


Figure 2.3: Two signals at a low receiving gain. ©2009 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "SDR implementation issues: RF front end nonlinearity and dynamic computing resource allocation," Software Defined Radio Forum, Washington DC, 2009.

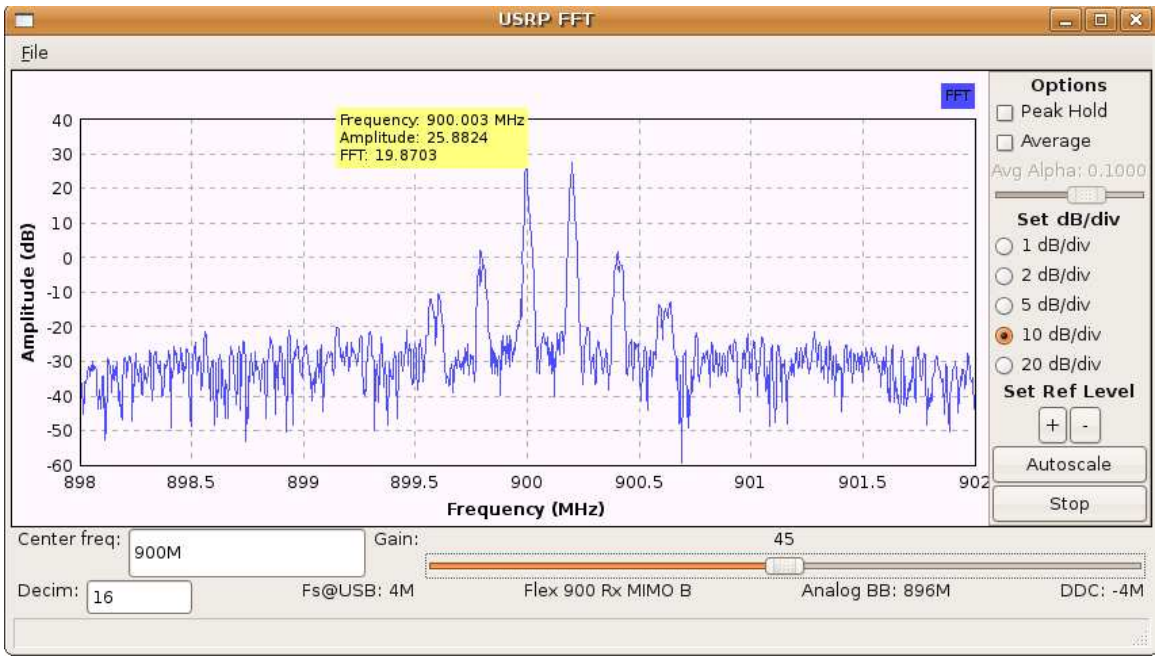


Figure 2.4: Inter-modulated signals at a high receiving gain. ©2009 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "SDR implementation issues: RF front end nonlinearity and dynamic computing resource allocation," Software Defined Radio Forum, Washington DC, 2009.

Adjacent Channel Interference

The author also studied the problem of adjacent channel interference (ACI). This is particularly noticeable when the receiver must simultaneously process multiple independent signals or when the weak signal is adjacent to a strong signal from a nearby transmitter. To demonstrate ACI, the author decreased one signal's power level 32 dB lower than the other signal, as shown in Figure 2.5. The author then moved the center frequencies of the two signals closer at a separation of 100 kHz as shown in Figure 2.6. We can see that the frequency selectivity of the receiver is degraded as the energy from the strong signal leaks into the bandwidth of the weak signal.

2.3.2 Nearby Channel Data Communication

An SDR can't set its receiver gain too high because of nonlinear effects as shown in Figure 2.4. However, if the receiver gain is too low, the received signal's SINR might be too low, therefore resulting in high BER and packet error rate (PER). Here the author used GNU Radio and USRP 1 (with a FLEX 900 daughter-board) to characterize PER variation against SINR. Instead of varying the receiver gain, the author changed the transmitter

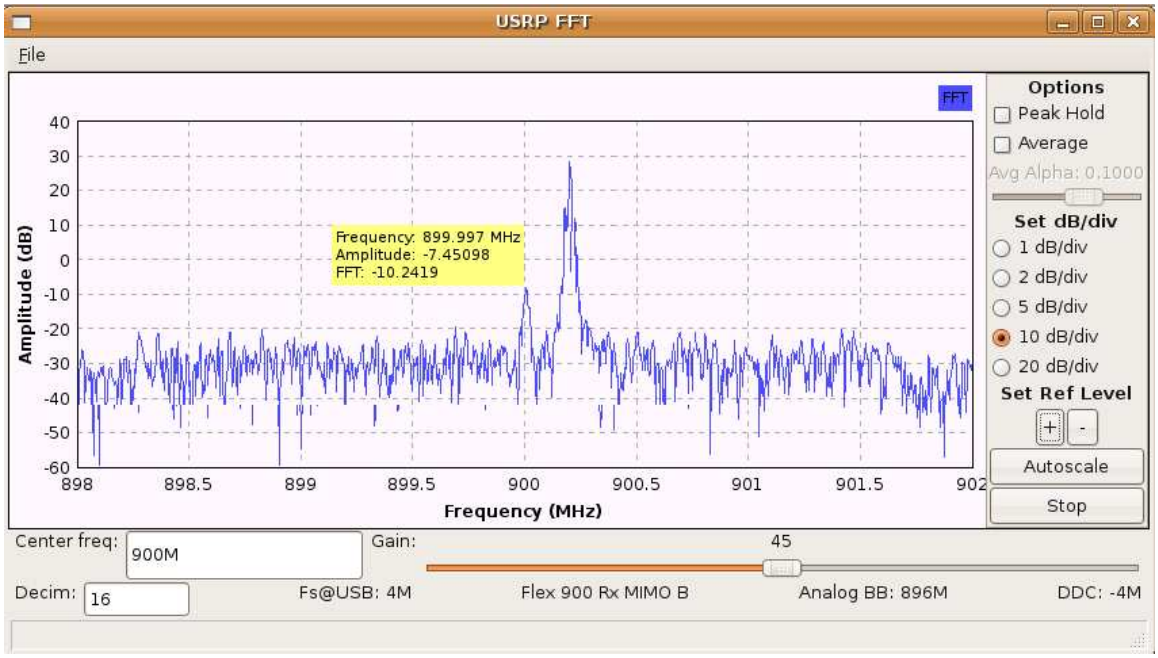


Figure 2.5: Two signals with different transmitting powers. ©2009 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "SDR implementation issues: RF front end nonlinearity and dynamic computing resource allocation," Software Defined Radio Forum, Washington DC, 2009.

power over a large range because it is a more appropriate way to vary SINR at the receiver. (The noise level remains constant.) The receiver gain is set one third of the maximum value at the PGA in Figure 2.2. In calculating PER, the receiver received 5000 packets and each one had an error checking header. Figure 2.7 shows the results and it also indicates the SINR where inter-modulation begins when the receiver is processing equally strong signals on adjacent channels.

The calculated SINR is determined by USRP's architecture because the signals used in calculation go through function components like filtering, amplification, and gain control. The absolute values of USRP-derived SNR measurements are known to be unreliable [55], but their relative values are consistent with each other and thus useful for purposes of comparison, as reported here. The same signal samples are also used in demodulation; therefore, Figure 2.7 does demonstrate that non-linearity and SINR significantly limit the proper operating range for received signals at different power levels. In this example, the effective dynamic range (as described here) of the receiver is only about 7 dB.

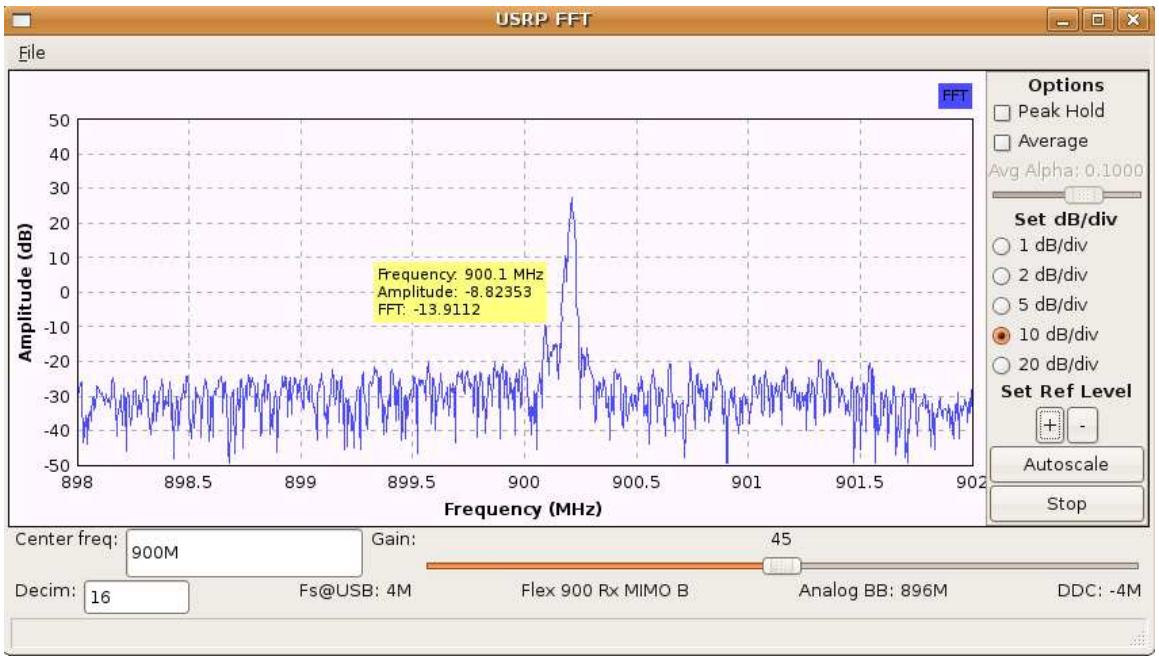


Figure 2.6: Adjacent channel interference. ©2009 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "SDR implementation issues: RF front end nonlinearity and dynamic computing resource allocation," Software Defined Radio Forum, Washington DC, 2009.

2.4 Dynamic Computing Resource Allocation in SDR

Fundamentally, any radio system must guarantee a real-time performance. From the perspective of network data communication, data are transmitted as segments at the PHY layer; to receive all the data, processing time for each data segment is limited – determined by the transmitting data rate – since the radio must accept and process this data segment before the next segment arrives. Therefore, the overall signal processing in a radio system must have this deadline-driven constraint [56] – all signal processing functions must be completed within a certain time or radios may not function appropriately.

Following the receiver path chain, the antenna and other analog parts can process signals of any bandwidth at an almost instant speed. Therefore, the real-time performance constraint mainly lies in the digital part. After the ADC, a minimum amount computing resources, measured for example, by computing cycles and memory space, is needed to process the digital samples in achieving a needed real-time speed that matches the incoming digital samples' speed. Different waveforms require different amounts of computing resources; for example, 16 QAM requires more computing resources than BPSK in achieving the same symbol rate. In conventional digital radios, the analog-to-digital conversion

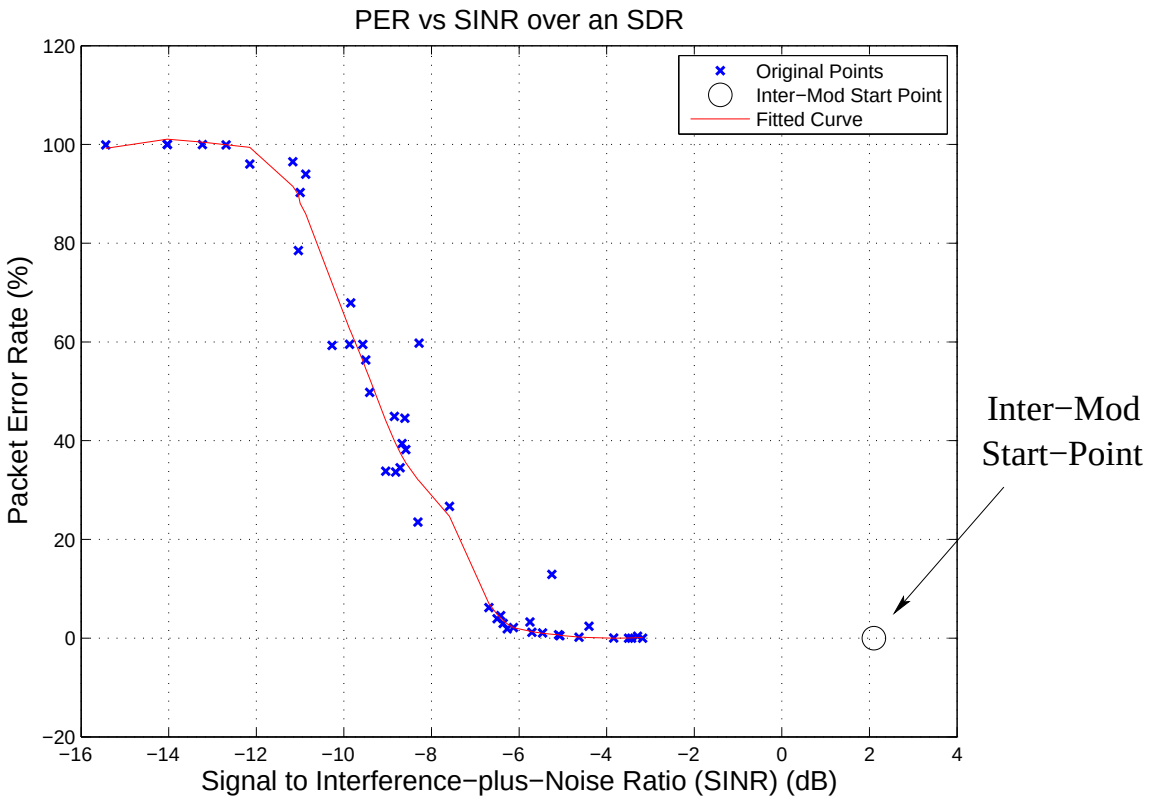


Figure 2.7: PER vs SINR distribution. Note that the absolute values of SINR are low because of known problems with the USRP/GNU Radio measurement systems, but the relative values and the illustrated functional dependence are correct. ©2009 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "SDR implementation issues: RF front end nonlinearity and dynamic computing resource allocation," Software Defined Radio Forum, Washington DC, 2009.

occurs either at the baseband or at a low IF band, which dramatically reduces the speed of the digital signal samples. Therefore, the baseband data rate is usually at the same order of the sample rate after ADC. Thus, the real-time performance requirement of conventional digital radios can be equally determined by the baseband data rate.

Therefore, it is possible to achieve both real-time performance and low power consumption for a single digital waveform because (1) the RF radio signals can be down-converted to the baseband with a necessary minimum data rate, thus minimizing required computing resources; (2) the required digital signal processing functions can be fully optimized and executed on Application-Specific Integrated Circuits (ASICs). Most commercial radios are produced, (cell-phones for example) in this way. In commercial digital radios like a DSP based P25 radio, the computing resource requirement for all P25 radio tasks is thoroughly quantified, based on the selected waveforms and the baseband data rates to achieve

real-time performance. A DSP with required computing resources (in terms of CPU and memory), is dedicated usually with some leeway to the signal processing functions that correspond to such waveforms and data rates.

However, it is quite a different story for SDR. First, moving digital signal processing functions closer to the radio antenna requires a high data rate and more functional components in the software domain; this implies additional computing resources such as computing cycles in signal processing and memory space to hold more signal samples. Second, SDR usually needs to reconfigure itself to support more than one application; this makes it challenging to highly optimize computing resource allocation for dynamically configured radio functions. Third, developing and executing SDR systems requires a running computing system. Such a system consumes computing resources, maybe at a significant level, without even running any SDR code.

Among the three most popular computing systems for SDR development including FPGAs, GPPs, and DSPs – FPGAs are not flexible in reconfiguration and consume significant power, but they can support real-time performance [57]. GPPs are capable of reconfiguration but are power demanding. DSPs consume less power and can reconfigure for simple real-time tasks but are not able to support computationally intensive tasks [16]. To date, it is still an open challenge to design an SDR platform which is flexible while supporting real-time performance with a limited battery life.

Supporting real-time performance in a SDR system puts a stringent performance requirement on all the components including the RF front end, AD/DA converters, component interconnect interfaces, and computing devices. This section primarily focuses on the impact of computing resource allocation on the performance of GPP based SDRs. Next section investigates the overall execution latency in a SDR system.

2.4.1 Supporting Real-time Performance in SDR

SDR is quite different from conventional digital radios in that it usually captures signals samples at a high IF frequency. The result is that the digital domain has to deal with a sample rate after the ADC much higher than the baseband data rate – sometimes several orders of magnitude higher. For example, a P25 radio may only require a data rate of 9.6 kB/s [58], but an SDR requires a sample rate of at least 1 MS/s if it captures signals with a bandwidth of 0.5MHz; this is about a 100 times higher data rate. Depending on the type of digital signal processing functions after ADC and before the baseband, processing such a much higher sample rate in the digital domain may require more computing resources than a specific waveform at the baseband. This is certainly true for required memory space

and its associated memory operations. Therefore, the real-time performance requirement of SDR is determined by both the IF band sample rate and the baseband data rate for a specific waveform. For simple waveforms, processing the IF band sample rate requires far more computing resources than the baseband data rate in achieving the required real-time performance and dramatically increases the associated power consumption, which fundamentally limits SDR's applications.

Further, unlike conventional digital radios which often dedicate different optimized computing devices to different parts of a system, SDR accommodates many digital signal processing functions in a single computing domain and targets many radio applications. Not only does this limit the overall system optimization, but also it complicates the system performance analysis because all running functions compete with each other for computing resource. Such competition is particularly complicated in GPPs and DSPs because CPU and memory consumption for multiple functions can be non-linear in most operating systems. This creates a big challenge in analyzing CPU and memory requirements for allocating resource dynamically in achieving real-time performance. While FPGAs can, in theory, allocate computing resources – the programmable gates – in a linear manner for different functions, the resource usage cannot be optimized, resulting in wasteful power consumption.

2.4.2 Experimental Investigation of Dynamic Computing Resource Allocation in SDR

In this section, the author uses a concrete example to illustrate the impact of computing resource allocation on the real-time performance requirement in GNU Radio (version 3.1 in this example). As shown in Figure 2.8, the author set up a direct radio link between two nodes by continuously transmitting a fixed amount of data from the transmitter to the receiver using GMSK. The data is transmitted in fixed-length packets at a fixed rate. In the receiver path, the USRP sends digital samples at an IF band to its connected GPP. In the experiment, the sampling rate fed to the GPP is fixed at 4 MS/s and the baseband data rate is 50 kB/s with 2 samples per symbol. The samples are then low-passed filtered before demodulation. The final baseband data is saved in a message queue. All the function blocks in Figure 2.8 are implemented in terms of signal processing blocks which require CPU cycles and memory space to do their work. More details about how GNU Radio works is available in [46]. To analyze the relationship between computing resource allocation and real-time performance, the author also adds a variable number of copies of the same low-pass filter after the USRP, but feeding their output samples to a null block which simply

dumps the data. The low-pass filter is a finite impulse response filter using Hamming window with 1241 taps. The used GPP is an Intel dual-core CPU (model T9300) with a frequency of 2.50GHz. It has 3.5 GB of memory.

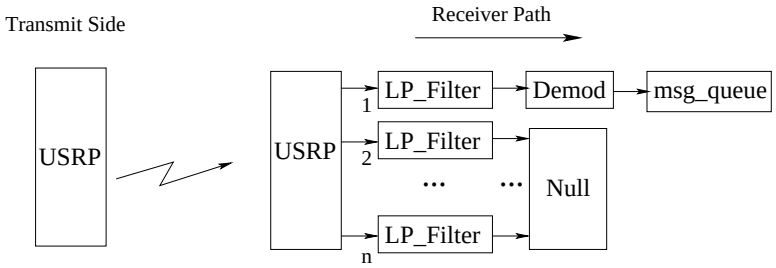


Figure 2.8: Experiment set up. ©2009 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "SDR implementation issues: RF front end nonlinearity and dynamic computing resource allocation," Software Defined Radio Forum, Washington DC, 2009.

Experiment Results and Analysis

In this experiment, the receiver has a basic signal path including the signal processing blocks of USRP, demodulation, and only one low pass filter. It needs to receive 5000 packets; the overall computing resource consumption in receiving those packets and the PER are both calculated. The author then added a variable number of extra low-pass filters as shown in Figure 2.8 and measured both the computing resource consumption and PER accordingly. Specifically, the author used four methods in estimating the computing resource consumption. The package SYSSTAT [59] and Oprofile were used to measure the overall user domain and individual GNU Radio function CPU utilization; the Python Profiler [60] was used to measure the overall CPU time to process all received packets, and the Linux system monitor was used to measure the memory consumption. The Python Profiler uses deterministic profiling by precisely timing the intervals between events like function calls, function returns, and exception events [60] while SYSSTAT and Oprofile use statistical profiling by randomly sampling the effective instruction pointer and deducing where time is being spent. To minimize the OS impact, the author restarted the computer in a clean environment for each running case.

Figure 2.9 shows the variation of overall CPU consumption in the user domain, including both GNU Radio and other running system programs, as the number of extra low pass filters increases. The CPU consumption is around 2% of all available CPU cycles without running GNU Radio. Though not shown in the figure, the author also found from the Linux system monitor that the memory utilization remained almost constant around 397MB (11%

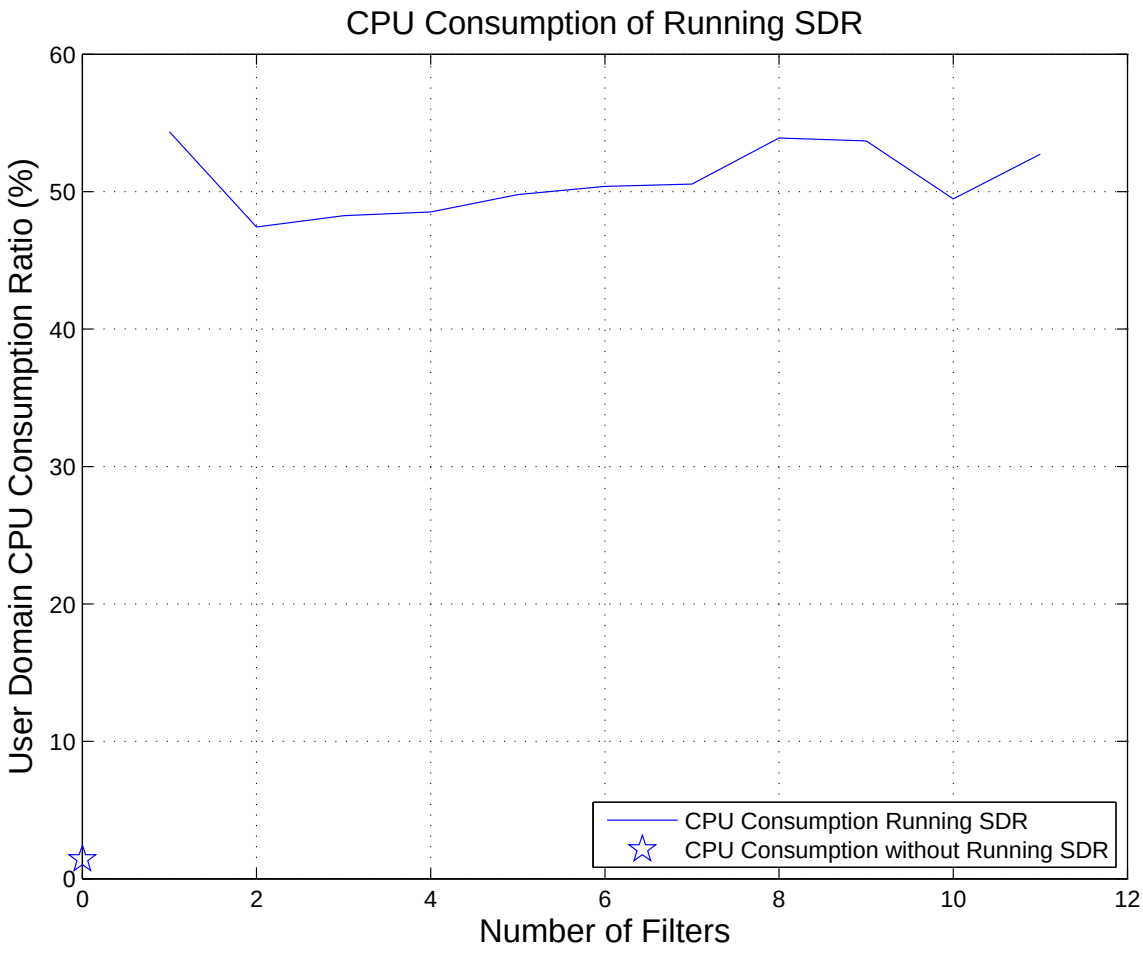


Figure 2.9: User domain overall CPU consumption variation. ©2009 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "SDR implementation issues: RF front end nonlinearity and dynamic computing resource allocation," Software Defined Radio Forum, Washington DC, 2009.

overall utilization) when running GNU Radio and 373 MB (10.5% overall utilization) when not running GNU Radio. Therefore, memory consumption had a negligible impact on the experiment.

As we can see from Figure 2.9, no trend seems to exist purely for the overall user domain CPU utilization. There may be multiple reasons to explain this. Fundamentally, GPP's CPU has limited hardware resources, for example, registers and ALUs. They are shared among different tasks as the OS assigns resource and CPU clocks for each running task. There is variation for the overall system CPU utilization and such variation may obscure the CPU increment introduced by extra running filters solely in GNU Radio.

To further investigate the variation of computing resource consumption, the author then

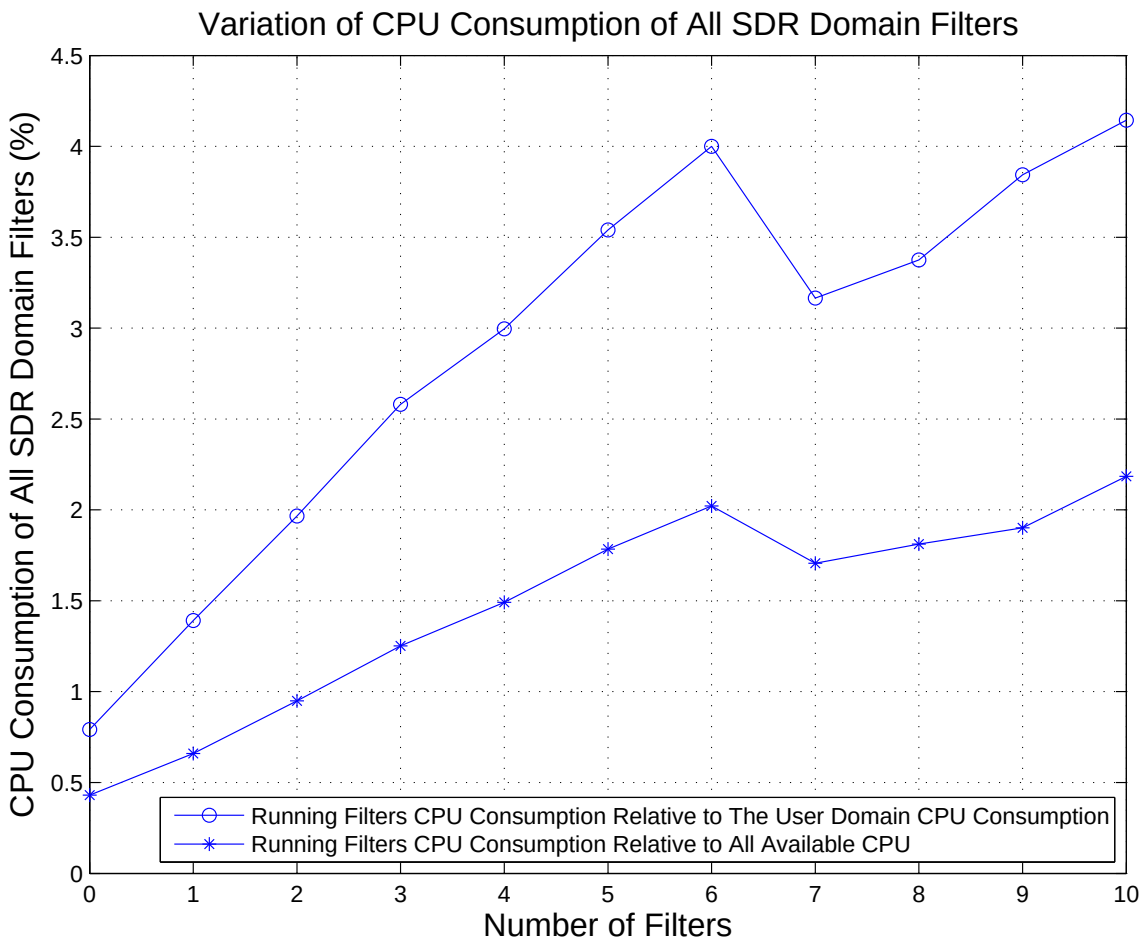


Figure 2.10: CPU consumption variation of different number of low pass filters. ©2009 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "SDR implementation issues: RF front end nonlinearity and dynamic computing resource allocation," Software Defined Radio Forum, Washington DC, 2009.

specifically analyzed the CPU consumption of the low pass filters by using Oprofile because the filter operation was the main CPU consumption increment among all running cases. As shown in Figure 2.10, the CPU consumption of all running low pass filters increases linearly to a total of 7, and then drops at 8, followed by a continuous increase afterward. Two lines are shown in Figure 2.10, one is the CPU percentage of all running filters against the user domain overall running programs, the other is the CPU percentage of all running filters against all the available system CPU. We can see that each extra filter only consumes about 0.3% of the overall CPU; that result confirms the author's estimation in the previous paragraph. Because it is now confirmed that the required CPU increases linearly at least for the first 7 running cases, next the author studied the impact of CPU consumption increase

on SDR performance – here the PER. As shown in Figure 2.11, the PER for the first 7 running cases is, in almost all cases 0, then it increases abruptly to almost 100%.

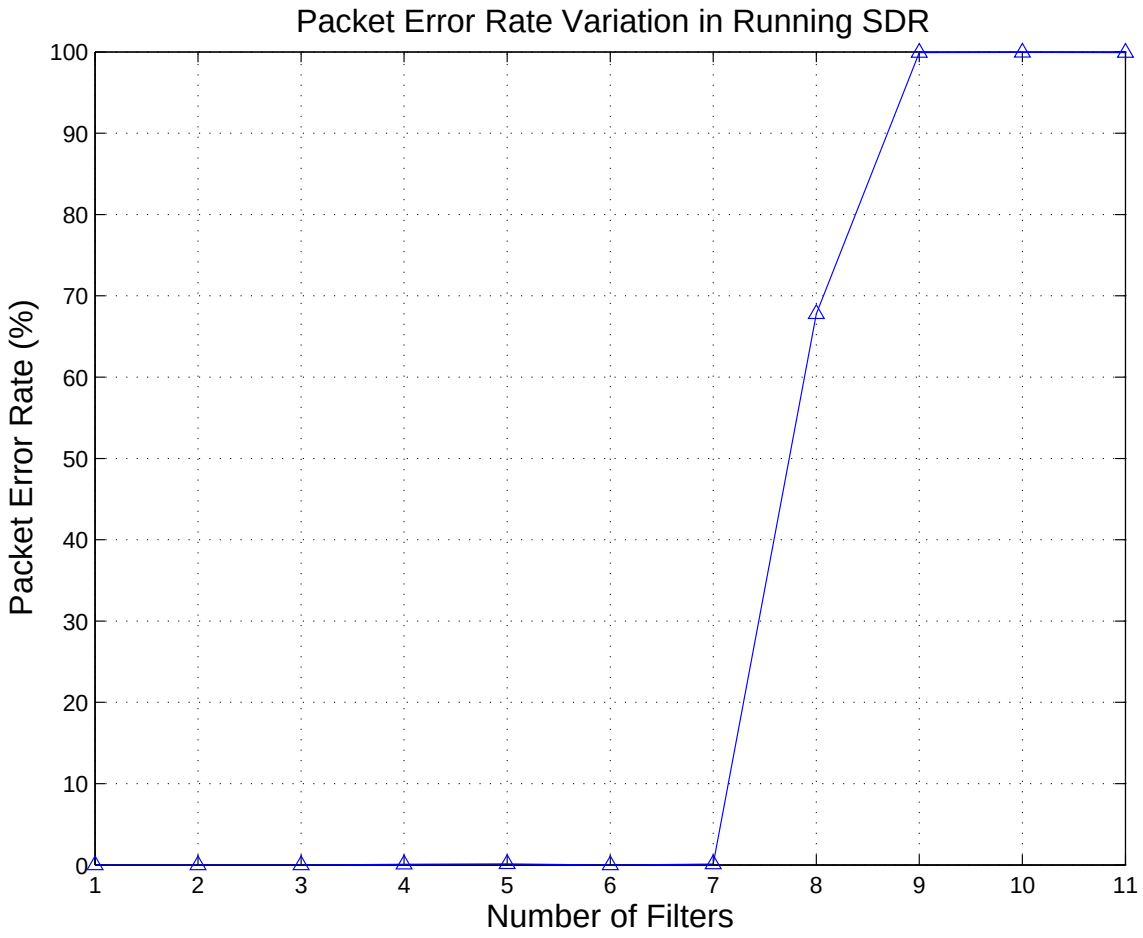


Figure 2.11: PER distribution as the number of low pass filters increase. ©2009 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "SDR implementation issues: RF front end nonlinearity and dynamic computing resource allocation," Software Defined Radio Forum, Washington DC, 2009.

To further investigate the CPU consumption, the author also used Python Profiler to measure the overall timing requirement in receiving 5000 packets. The author found that the required time was around 96.5 seconds for all the first 7 running cases, and then it increased sharply for following cases, as did the PER correspondingly. The author also found that the receiver path, starting from running case 8, experienced a significant number of USRP overruns (USRP samples were dropped because they were not read in time by the following signal processing block.) In such situations, the transmitter actually needed to send more data packets so that the receiver could receive up to 5000 packets. Therefore,

the CPU time increase indicated the increase in the number of dropped signal samples from USRP.

Given the above data results, we now can understand why the CPU percentage of all running filters in Figure 2.10 doesn't increase linearly after the number of 7. GNU Radio execution slows down if signal samples are dropped once a while and the CPU consumption dynamics thus changes. But most importantly, at a threshold value, a small amount of required CPU increment may result that an SDR receiver performance deteriorates abruptly because it cannot allocate enough computing resources to processing the incoming digitized samples at the required real-time speed.

2.5 SDR Execution Latency

A wireless network performance is determined by many factors; among them, the MAC layer is especially crucial because wireless medium is shared among all nodes and they may interfere with each other. Further, each node may experience shadowing, fading, and multipath propagation. As a result, a wireless network's PHY layer no longer has a constant channel capacity, in contrast with wired networks [61].

SDR achieves multi-band multi-mode reconfigurability by moving digital signal processing functions progressively closer to the radio antenna and using software methods to replace analog functions. As discussed in Section 2.4.1, however, doing so also means that the computing device in a SDR system must process a much higher data rate than the baseband data rate of a signal waveform. Further, significant amount of data must be sent to the computing domain. Because of the memory hierarchy in a GPP, the peripherals connecting the RF front end to the GPP, multiple data buffers along the radio transceiver signal processing, and computing resource contention in the OS, GPP-based SDR have performance challenges not only in enabling real-time digital signal processing, but also of supporting low latency execution of PHY and MAC layer functions. This section explores most latency sources by following the entire receiver chain from the RF front end to MAC functions in software domain. It also gives numerical latency measurements taken by using times-tamps.

2.5.1 Latency Sources in GNU Radio

If we follow the receiver chain in GNU Radio, there are six factors that may introduce latency, as shown in Figure 2.12:

1. analog signal processing and wire delay in USRP's analog circuits,

2. sampling delay in converters and programmable gain amplifier (PGA),

3. filter processing time in FPGA,

4. USB's data queue transmission mechanism, and its limited buffer size and transmission speed,

5. GNU Radio's streaming architecture in both radio control and PHY information, signal processing, and GNU Radio scheduler, which will be detailed in later sections,

6. GPP operating system (OS) latency and uncertainty, GPP memory hierarchy, and others.

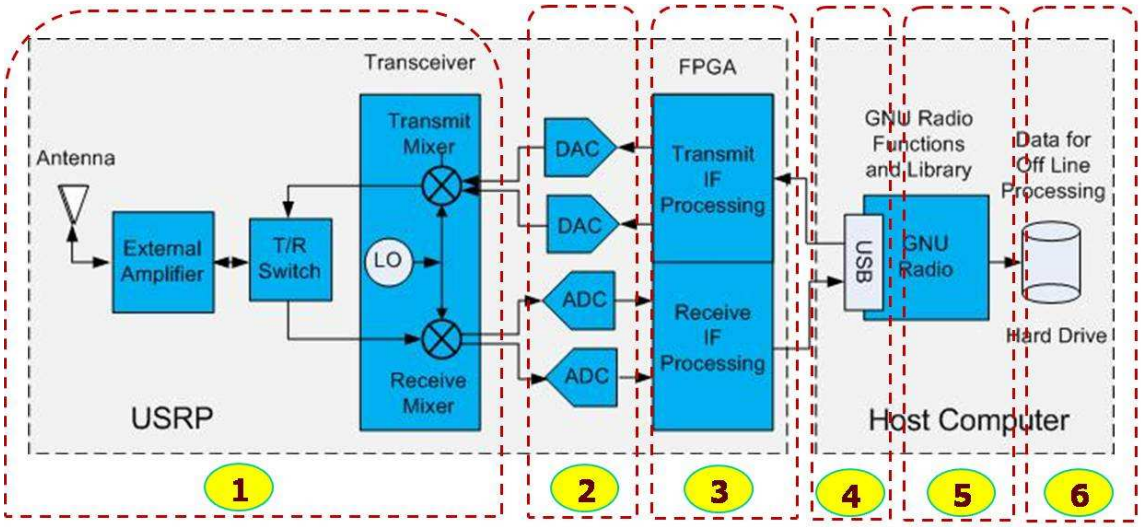


Figure 2.12: Decomposition of a SDR system consisting of GNU Radio and a USRP. ©2008 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "Software defined radio execution latency," Software Defined Radio Forum, Washington DC, 2008.

Let's use Δ to represent latency in different resources; the system total latency can be decomposed into into four parts:

$$\Delta = \Delta_{\text{USRP}} + \Delta_{\text{USB}} + \Delta_{\text{GNU Radio}} + \Delta_{\text{GPP}} \tag{2.1}$$

where Δ_{USRP}, Δ_{USB}, $\Delta_{\text{GNU Radio}}$, and Δ_{GPP} are the total delay time introduced by USRP, USB, GNU Radio software, the host GPP. Individually, Δ_{USRP}, $\Delta_{\text{GNU Radio}}$, and Δ_{GPP} can be further decomposed as shown below:

$$\Delta_{\text{USRP}} = \Delta_{\text{Analog}} + \Delta_{\text{Wire Delay}} + \Delta_{\text{AD/DA}} + \Delta_{\text{FPGA}} + \text{others}$$
$$\Delta_{\text{GNU Radio}} = \Delta_{\text{Signal Processing}} + \Delta_{\text{Scheduler}}$$
$$\Delta_{\text{GPP}} = \Delta_{\text{OS}} + \Delta_{\text{Memory Operation}} + \text{Others} \quad (2.2)$$

The following sections briefly analyze each of the four parts in Eq. 2.1 separately.

Execution Latency in USRP

As shown in Figure 2.13, the USRP introduces latency because of analog signal processing and wire delay, TX/RX switch, tuning in the voltage-controlled oscillator (VCO), signal sampling process, programming delay in the programmable gain amplifier (PGA), and filter processing in FPGA. The TX/RX switch and VCO do not impact steady state execution latency, and most other parts' latency is negligible with the typical value of 1 μs in the FPGA.

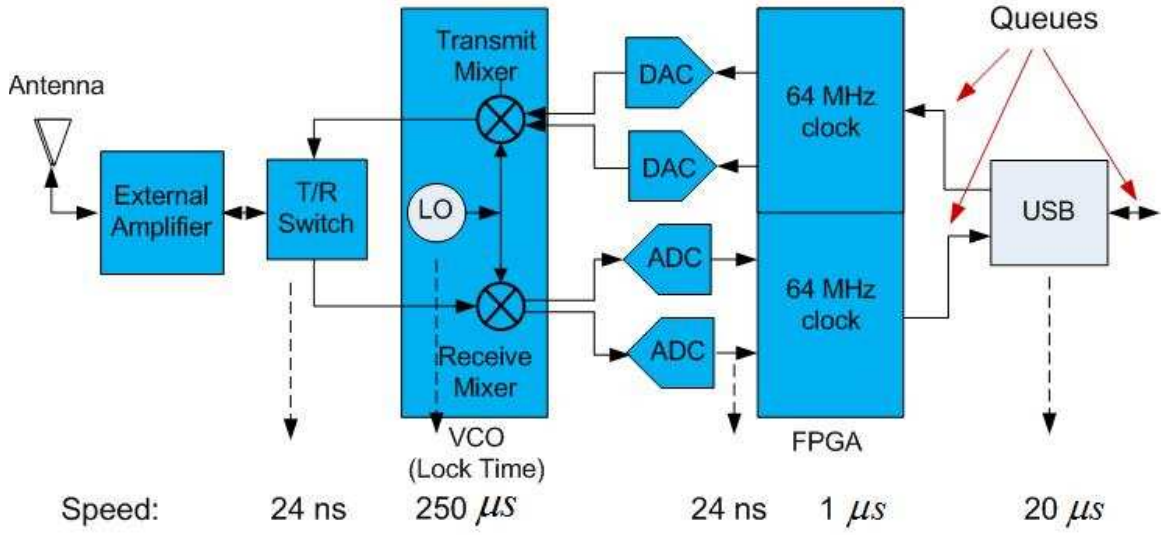


Figure 2.13: the time scale at each radio component. ©2008 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "Software defined radio execution latency," Software Defined Radio Forum, Washington DC, 2008.

USB Characteristics

USRP I connects to the host computer through USB 2.0 ports with buffers on both sides: 8 kB on the USRP side and 32 kB on the host computer side. All the data is put into

queues before it goes through the USB connection, as shown in Figure 2.13. The USB connection transmits data in blocks with a minimum size of 512 B. This means that smaller data chunks will have to wait for incoming data to fill up to 512B before they can pass the USB. Therefore, the USB latency, which is the time duration for moving data from FPGA to the USB driver on the PC, is decided by three factors: packet size, USB data rate, and buffer size on both sides of the USB connection. The USB data transmitting latency is

$$\Delta_{\text{USB}} = \frac{f(512, \text{fusb-nblocks} \times \text{fusb-block-size})}{\text{sample-size} \times f_s} \tag{2.3}$$

where $f(x, y)$ depends on the amount of data in the buffer and is at least x and at most y, fusb-nblocks is the number of data blocks in USB, fusb-block-size is each block's size, f_s is the sampling frequency, and sample-size is usually 32 bits for complex signal samples.

Because of the queue structure in USB, there is latency variation among different data rates in the signal flow. For burst signals flowing from the RF front end when both buffers are empty, the latency is decided by Eq. (2.3), which also determines latency for low rate continuous signal flow. For high rate continuous signal flow, new arriving signal data have to wait for previous data flowing out of the buffer. Therefore, the latency accounts for the time that buffers empty previous data. For the TX chain, it is quite similar except that the PC side memory buffer size is larger.

GNU Radio Running Mechanism

Programming in the GNU Radio platform uses a combination of C++ and Python, a simple, high-level programming language. The computationally intensive processing blocks are implemented in C++ while the control and coordination of these blocks for applications that sit on top are developed in Python. Starting with the version 3.2, Python is optionally used in run GNU Radio applications.

When executing SDR functions, the GNU Radio scheduler processes data as a stream of homogeneous items in any active flow graph. It breaks each data stream into chunks and feeds these chunks one at a time into each block in the flow graph using the mechanism described in Algorithm 1 [62].

The scheduler scans through all included signal processing blocks in the flow graph from the head to the end, and then loops back. The scheduler is essentially a cyclic poller, calling each block in turn to perform its processing function, always cycling in the same order. This mechanism creates extra latency in looping. Even if one block has the highest priority for a particular task, it must still wait for following and preceding signal processing blocks.

Algorithm 1 GNU Radio's core working mechanism.

```
while flowgraph.start do
  for i = 0, ··· , total number of blocks do
    if block i has enough input data and sufficient memory for output port then
      process block i
    end if
  end for
end while
```

GPP Memory Hierarchy and OS Environment

In a GPP operating system environment, GNU Radio executes all the PHY layer functions including IF band and baseband functions. GPPs' memory hierarchy and OS's characteristics will bring overhead that does not arise in FPGAs and ASICs.

Specifically, modern GPP OS is designed to maximize resource utilization – to assure that all available CPU time, memory, and I/O are used efficiently, and that no individual user takes more than her fair share [49]. To maximize resource utilization, the OS uses processes and threads to run multiple jobs simultaneously: applications, system programs, drivers, etc. The CPU scheduler manages all threads and processes according to scheduling algorithms like First-Come, First-Served (FCFS) scheduling, or Round-Robin (RR) scheduling. All scheduling algorithms balance several criteria like CPU utilization, throughput, waiting time, and response time. The implication for GNU Radio is that the OS will create latency uncertainty for GNU Radio implementation because GNU Radio signal flow graph processing is just one process running in the operating system even though GNU Radio can set a high priority for this single process.

Furthermore, most GPPs still follow the von Neumann architecture which has a memory hierarchy as shown in Figure 2.14. GPPs run any program instructions (with data) in the CPU with registers while instructions and data are usually stored in the disk or memory. There is a vast speed difference between CPU and memory [48]. To solve this problem, several levels of caches are inserted between the CPU and memory, and speculative methods are used to pre-fetch instructions or data into the caches. However, any failure in speculative pre-fetching will cause the CPU to wait for data read from memory with a long delay, as shown in Figure 2.14.

2.5.2 Latency Measurements

Section 2.5.1 analyzes the possible latency, while this section will give numerical measurements. To measure the latency, the author used two USRPs connected to one computer so

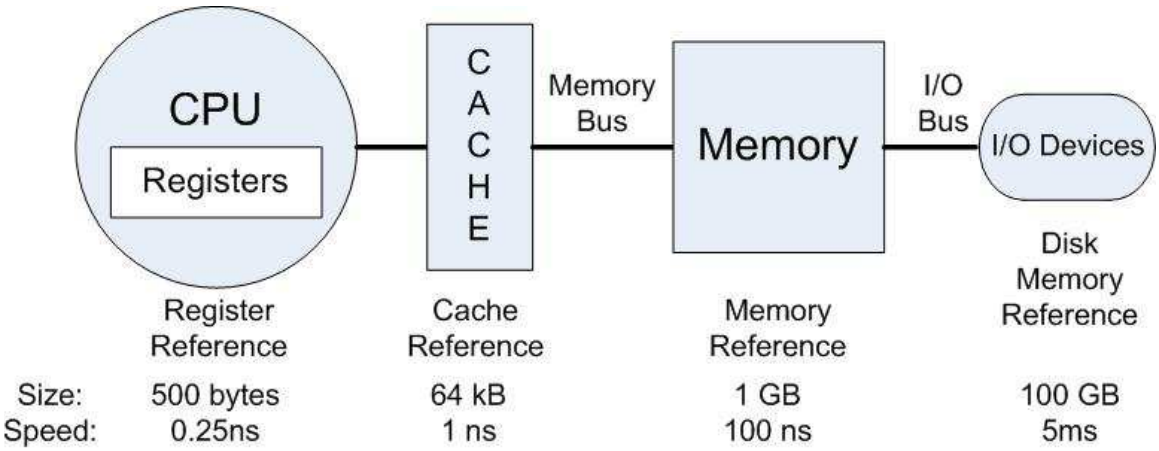


Figure 2.14: Memory hierarchy in GPPs and the speed difference. ©2008 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "Software defined radio execution latency," Software Defined Radio Forum, Washington DC, 2008.

that times-tamps can be used within one computer to measure the time between transmitting a packet and receiving it. As shown in Figure 2.15, the author implemented a digital transmitter and receiver – each in its own USRP – and four times-tamps were inserted: the first (Timer Head) is right before the data packetizing, the second (Timer 1) is right before the transmitter IF band, the third (Timer 2) is right after the receiver's IF band, and the fourth (Timer Tail) is right after data de-packetizing. The experiment used an Intel Core 2 dual-core processor (@2.40 GHz) with 2GB of memory and the GNU Radio code was based on version 3.1.

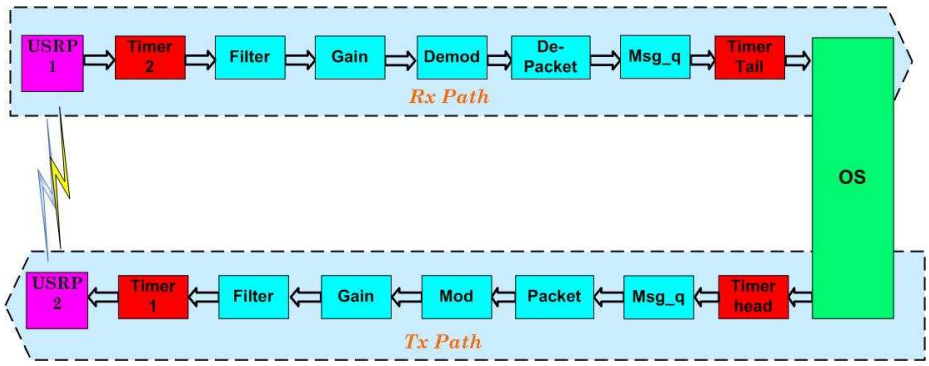


Figure 2.15: Experiment setup for measuring SDR execution latency. ©2008 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "Software defined radio execution latency," Software Defined Radio Forum, Washington DC, 2008.

Next, the author transmits different sizes of packets and measure the total latency, i.e., the time difference between Timer Head and Timer Tail. For BPSK at different bit rates, the latency time vs. packet size is shown in Figure 2.16.

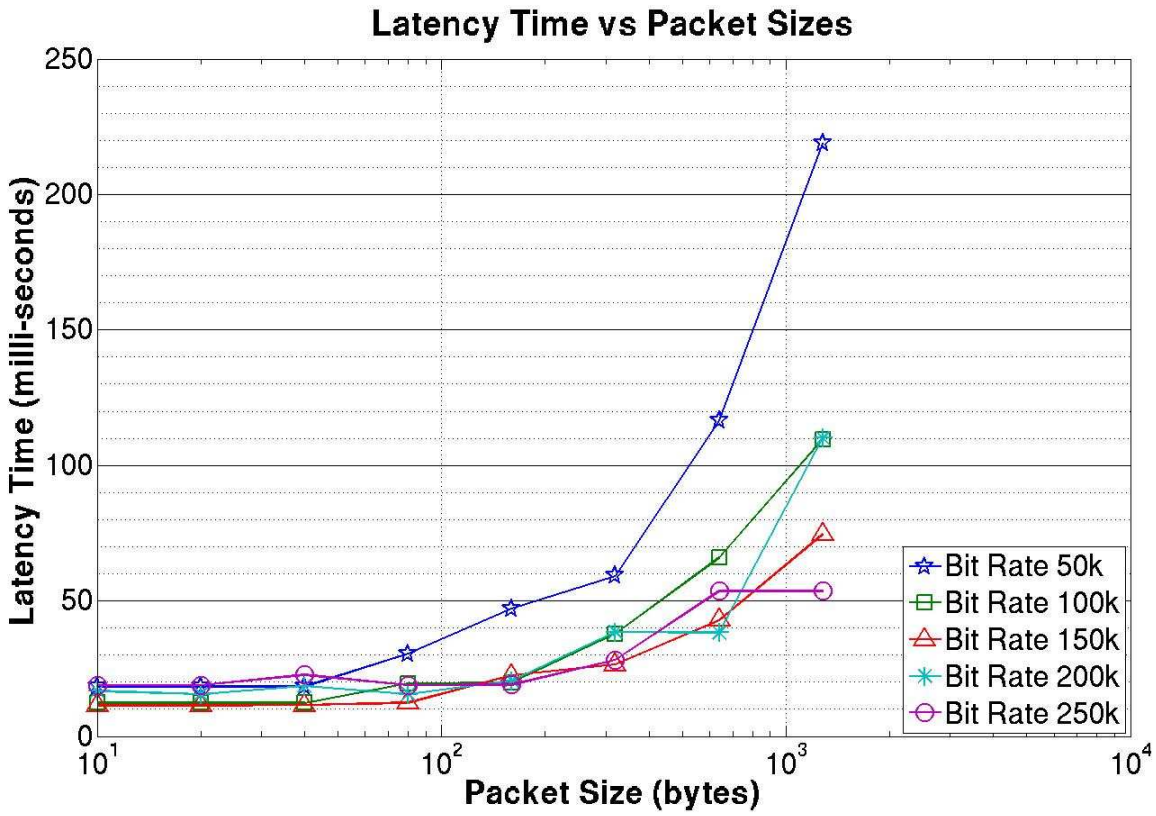


Figure 2.16: The latency time between transmitting a packet and receiving it using BPSK as a function of packet size and bit rate. ©2008 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "Software defined radio execution latency," Software Defined Radio Forum, Washington DC, 2008.

As we can see in Figure 2.16, the latency time is on the order of ten milliseconds between sending and receiving even a small packet (10 bytes) at all test bit rates. As the packet size increases beyond 100 bytes, this latency time increases continuously and reaches the order of 100 milliseconds for 1000 bytes, though a higher bit rate results in a shorter latency.

Following Algorithm 1, a big packet in GNU Radio is broken into a sequence of data chunks and each one is processed following the flow graph shown in Figure 2.15. Considering the uncertainty from the three queues shown in Figure 2.13 and the limited size of USB buffers, the author further measured the amount of time spent on transmitting and receiving a packet at different modulations and bit rates. Particularly, the author let the transmitting packet go directly to the receiving side at the baseband and USRPs were not used, therefore removing the impact of waiting for emptying USB queues and other hardware latency. Such time latency is purely decided by the GNU Radio architecture and the CPU speed, and theoretically it is proportional to the number of samples (per second) for

the same baseband functions setting if there is enough available CPU and memory.

As an example in Figure 2.17, the author used BPSK in transmitting one packet and receiving it at the baseband. The behavior of time latency vs. packet size is similar to Figure 2.16, though it is one order of magnitude smaller. Such latency is usually at the scale of tens of micro-seconds for small packets.

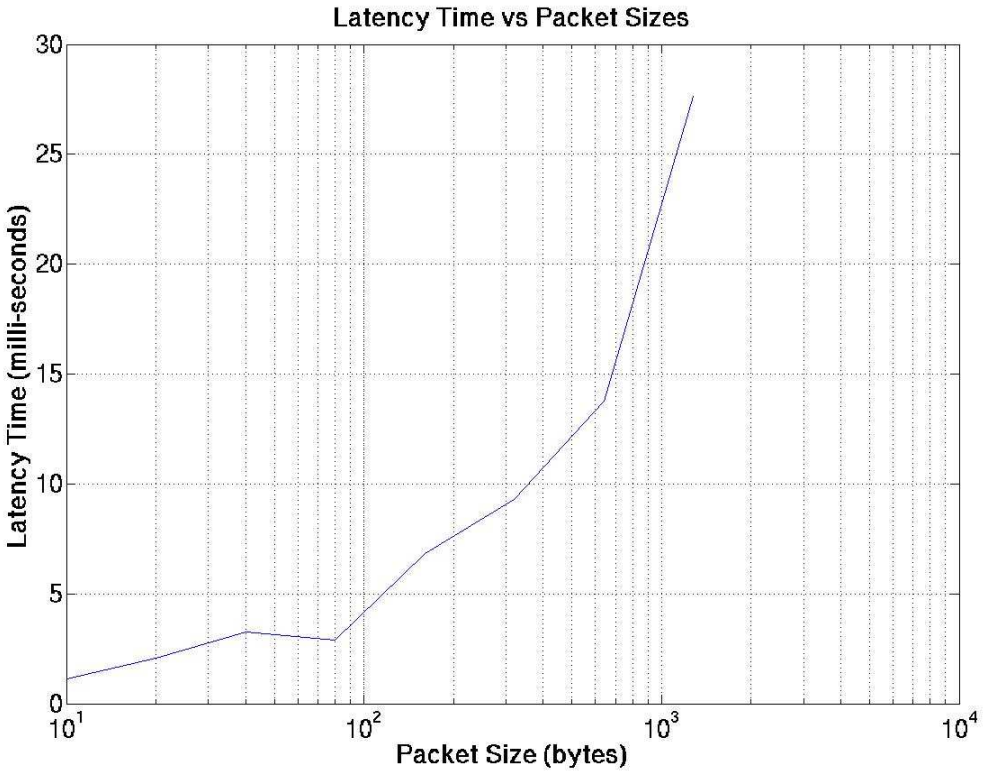


Figure 2.17: BSPK baseband signal-processing time. ©2008 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "Software defined radio execution latency," Software Defined Radio Forum, Washington DC, 2008.

2.5.3 A Fundamental Latency Resource: Pipeline vs Sequential

Fundamentally, GPPs can only run one task at a time and each program is executed sequentially, even though some instruction and data level parallelisms like instruction pipeline, super-scalar instruction execution, and SIMD, are widely implemented [48]. The CPU scheduler switches CPU and memory resource among multiple running tasks quickly [49]. Even worse, GNU Radio scheduler adopts a purely sequential way to execute the signal flow from IF band to the MAC layer functions. Analog radio components, ASICs, FPGAs,

and commercial wireless devices all execute signals in parallel/pipeline (a continuous sequence of signals is executed simultaneously by a sequential set of components.). ASICs usually run multiple function components together, and thus are in a pipeline mode. FPGAs have the advantage of dividing different slices into different functions, therefore, executing signal processing functions in a pipeline mode. But, fundamentally, GPPs can only run one task at a time [49], even though some instruction and data level parallelisms like instruction pipeline and super-scalar instruction execution are widely implemented [48]. As a hypothetical example illustrated in Figure 2.18, the speed difference between pipeline and sequential is usually significant.

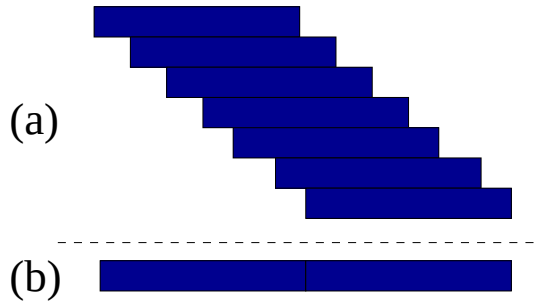


Figure 2.18: A hypothetical illustration of speed difference: pipeline (a) is 7 times faster than sequential (b). ©2008 Software Defined Radio Forum. Reprinted, with permission, from Ge et al., "Software defined radio execution latency," Software Defined Radio Forum, Washington DC, 2008.

Conventional radios (analog or digital) use a set of analog and digital components for specific radio tasks. Following an entire receiver signal processing chain, signals can continuously flow through all the analog components with negligible delay for signal processing; all the analog components process signals in a pipeline way. Usually ADC converters need limited sampling time to get digital samples (ADCs can easily work at the order of hundreds of MHz). All of the remaining signal processing like filtering, modulation and demodulation, source coding, and channel coding is done in the digital domain. For narrow band signals, the computational requirement is not tight; therefore, both DSPs and FPGAs can be used. For example, most public safety radios only use DSPs.

Nevertheless, it is usually challenging for a single DSP or FPGA to support wide-band waveforms, even for baseband signals. Commercial radio chips (ASICs) combine multiple computing components (system-on-chip) so that they can process multiple functions at the same time (parallel/pipeline). Certainly such a design can also get rid of latency in moving signals between different components. With an ASIC, the overall latency in the digital domain is therefore dramatically reduced. As an example, a typical Wi-Fi card [63] has several signal processing components for baseband PHY layer functions. They are used

for filtering, modulation, demodulation, MAC functions, encryption, and decryption. This architecture essentially executes signal processing in a pipeline mode.

Such architecture guarantees strict timing requirements, which is critical to IEEE 802.11's success. More specifically, there are several functions like TDMA (sync), CSMA (DIFS, SIFS), carrier sense, dependent packets (ACKs, RTS), fine-grained radio control (frequency hopping), etc., as shown in Table 4-1 that require precise and fast timing performance. A Wi-Fi chip must finish all the PHY/MAC layer functions within a few microseconds at both the transmitter and receiver side as well as doing all of the analog RF signal processing.

As a GPP based SDR example, GNU Radio executes all the PHY/MAC layer functions in a sequential way. Comparing Figure 2.16 and Table 2.1 (derived from [64]), we can see there are two to three orders of magnitude speed difference between GNU Radio and a Wi-Fi chip even though GNU Radio only executes narrow band signals while the Wi-Fi chip works on wide-band waveforms. As shown in Figure 2.17,considering even only baseband modem functions, sequential execution of PHY layer functions takes on the order of tens of micro-seconds for a small packet. The author believes that the most fundamental source for such a performance gap is parallel/pipeline vs. sequential signal processing even though the GPP's architecture does introduce some additional latency overhead.

Table 2.1: Summary of important timing constants in IEEE 802.11b, IEEE 802.11a, and IEEE 802.11g. (Please notice that the timing is based on performance of Wi-Fi ASICs.)

Parameter	Value			
	802.11b	802.11a	802.11g only	802.11g + legacy
SLOT	20 μs	9 μs	9 μs	20 μs
SIFS	10 μs	16 μs	10 μs	10 μs
DIFS	50 μs	34 μs	28 μs	50 μs
PHY layer PHY layer	192 μs [long] 192 μs [short]	20 μs	20 μs	20 μs

2.6 Implications for SDR design

Section 2.5.3 uses IEEE 802.11 as an example to illustrate timing requirements in wireless standards. Since execution latency is decided by the size of computational requirement and the available computing capacity, the author surveyed existing wide-band waveforms' data rates and summarize their computational requirements for baseband signals.

The complexity of algorithms used in telecommunications to reduce the bit error rate and increase spectrum efficiency has increased continuously. For example, multi-input

multi-output (MIMO) and broadband techniques have been developed to efficiently utilize both radio spectrum and energy while supporting high data rates. More channel estimation and adaption algorithms are used to reduce inter symbol interface (ISI) problem. As shown in [65], many wireless standards have already been and will continue to be created in the near future. Table 2.2 shows some wireless standards, their bit rates, and computational complexity at the baseband. Usually, higher data rates demand higher computing capacity for a same waveform. Algorithms used in smart antenna and traditional coding schemes certainly increase the computational complexity for SDR development.

Table 2.2: Data rates of wireless standards.

Wireless Standards	Bit Rate	Computational Complexity (million instructions per second)
GSM ([66])	270.833 kb/s	100
IEEE 802.11a ([67])	54.0Mb/s ()	5000
CDMA2000 ([68])	1.28Mb/s	2000
WCDMA ([67])	3.84Mb/s	3000
TD-SCDMA ([67])	1.28 Ms/s	3000
OFDM-VBLAST with 4x4 MIMO (WiMAX or IEEE 802.11n) ([69])	216 Mb/s	9600

2.6.1 Proposed Solutions

As we can see from the above sections, SDR architecture has to achieve a similar ability in reconfiguring as GPPs have and a similar execution speed as highly optimized ASICs have. Therefore SDR should better utilize available computing resources, for example, increasing more parallelism especially considering the increasing computational requirement from future broadband wireless technologies. This is far more important than using the increasing computing ability, as, for example, from DSP and GPPs. Therefore, the author recommends that future SDR architectures proceed in two directions: hybrid architecture as shown in Figure 2.19, or a multi-core based parallel architecture.

Hybrid architecture with a control processor may contain an embedded GPP, a reconfigurable FPGA, and some auxiliary ASICs. This arrangement has several advantages for implementing an SDR. The control processor is both necessary to handle non-DSP functions like branch, control, and decisions and efficient enough to coordinate different computing tasks and. ASICs are used for widely accepted wireless standards like Wi-Fi, WiMAX,

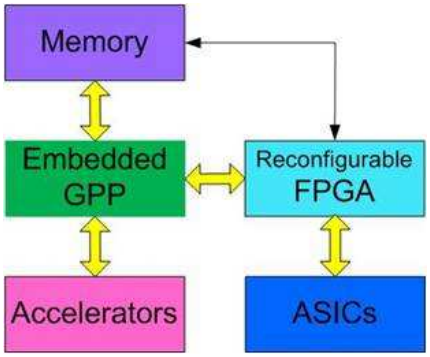


Figure 2.19: The proposed embedded GPP/FPGA hybrid architecture SDR

and LTE. Computation accelerators are for computationally intensive tasks such as video graphic processors. The FPGA can support most other PHY/MAC layer functions. Most importantly, some FPGAs support run- time reconfiguration by using techniques like Wires on Demand [70]. Therefore, upon reconfiguration request by the GPP, the FPGA is able to reconfigure quickly by using existing function bit streams, similar to the way that library functions are used for GPP programs. Such architecture is able to reduce the power budget by using low clock frequencies, thus saving die area and reducing static and dynamic power consumption. It can achieve parallelism, not only at the data and instruction level, but also at the task level by spreading tasks out among different computing components and different FPGA slices.

Another attractive approach to implement SDR is to use multi-core processor architecture. Because of the power wall, memory wall, and ILP (instruction level parallel) wall [48], single core based GPPs may not be able to increase computation speed significantly, and parallel architecture has became the dominate architecture in the industry. Multi-core architecture can use parallelism in achieving SDR's execution speed requirement while still maintaining the same level of flexibility as GPPs. For example, the Cell Broadband Engine (Cell BE) has nine heterogeneous cores [71], nVidia GPU has 256 cores [72], and Intel has an 80-core CPU [73]. Different signal processing functions can be executed on different cores in a pipeline mode. However, there are still some big challenges for both hardware and software architectures in parallel computing [48], and it is challenging to program in parallel.

On the other hand, high latency tolerant protocols can be used in networks built from SDR nodes with long execution latency [74]. The following is a non-exhaustive list of possible protocols, including changes to existing protocols [74], which could solve the problem: (1) TDMA: using a TDMA protocol would solve most of the latency problems, though it requires synchronization among the participating nodes. (2) Universal Header

Coding: based on a simple modulation scheme, headers and ACKs are the same through all the different protocols. This way, PHY layer functions can be implemented in hardware and thus a quick ACK or CTS response can be guaranteed. (3) Delayed ACK and a simple MAC: SDR execution latency is a big problem in the ACK reply and the RTS/CTS exchange. For the ACKs, we can delay them to a later point in time. We can also adopt a simple MAC like ALOHA protocol [75], instead of CSMA/CA, therefore, avoiding the RTS/CTS exchange.

2.7 Conclusion

This chapter investigates three performance issues in a SDR system through the examples of GNU Radio and USRP. Experimental results show that inter-modulation significantly limits SDR's receiver gain operation range. Further investigations conclude that general purpose processor (GPP) based SDR's performance is sensitive to computing resource contention. SDR performance may deteriorate abruptly if not enough computing resources are available due to the real-time constraint in radio function execution. Numerical measurements are given on the latencies within the software domain and the overall path from the transmitting packet to the receiving packet. Overall, GPP-based SDR suffers from execution latency because of delays in the memory hierarchy in the GPP, the peripherals connecting the RF front end to the GPP, and multiple data buffers along the radio transceiver signal processing chain as well as resource contention in the OS.

All the above three issues impact significantly performance of systems built from SDRs. From a research perspective, methods do exist to attack them. From a current user perspective, these issues must be migitated respectively through:

- The transmitting power and receiving gain of a SDR must be operated within the effective dynamic range of a SDR's RF front end.
- The computing resource requirement of all running applications must be thoroughly quantified under all possible execution situations and the SDR must provide sufficient resource for executing at least the maximum case.

These findings guided the design and development of our prototype DSA network.

Chapter 3

Signal Detection and Classification

The fundamental goal of DSA is to improve spectrum usage efficiency within three dimensions: frequency, time, and space, with a value even much higher than that achieved by current cellular systems [76]. The first and foremost requirement for any DSA implementation is to guarantee non-interference (or tolerable interference) to incumbent users in the above three dimensions. Therefore, spectrum awareness is a necessary component. Currently, there are three main types of spectrum awareness methods [77]: geo-locate and database lookup, spectrum usage beacons, and incumbent signal detectors. The last is the most popular. However, quickly and accurately detecting primary signals can be challenging. For example, the National Association of Broadcasters proposed to FCC that white space devices should accurately detect TV broadcasts at -116 dBm [78], which may be under the noise floor. In addition, the detection performance should hold up even under hostile conditions like shadowing, fading, and multi-path [42].

In general, spectrum sensing is required by a DSA network under two circumstances: (1) to identify spectrum holes, (2) to detect the return of primary signals on the channel being used by secondary users. The former case usually needs only to identify the presence of signals or not; while the later case must normally differentiate between primary signals and secondary signals. Signal modulation classification is widely used to differentiate signals.

This chapter first gives a general overview of signal detection and classification methods. It then focuses on a special signal modulation classification method – cyclostationary feature based signal detection. This method is attractive for detecting primary users because of its ability to distinguish between modulated signals, interference, and noise at low SNRs. However, a key issue of cyclostationary signal analysis is the high computational cost arising from the large number of required complex convolution operations. The author uses parallel computing on the Cell Broadband Engine (Cell BE) to attack this problem. The results are of general interest, even though it was not possible to implement the tech-

niques on the nodes that were available for us to use in the prototype DSA network.

3.1 Introduction

The goal of signal detection is usually to detect Signals of Interest (SOI) over a wide frequency range under conditions of low SNR, interference, and other dynamic wireless environments. It has important applications in the military domain where advanced techniques are required for real-time signal interception and processing. This is vital for decisions involving electronic warfare operations and other tactical actions like detecting improvised explosive device (IED) signal and jamming hostile signals. For DSA applications, signal detection mostly occurs in a band limited domain. In a digital domain, this means that a finite length of a high frequency signal is first down-converted into some IF band, and then numerical methods are used to discover the presence of any signals. Suppose that x is the sequence of collected data, the detection problem (with only AWGN) can be formulated through the hypothesis test:

$$\begin{aligned} H_0 &: x = n \\ H_1 &: x = hs + n \end{aligned} \tag{3.1}$$

where s is the unknown signal to be detected and n is an AWGN, h is the channel filter determined by the radio environment. There are two types of errors [79]: (1) false positive means the error of detecting a signal where none is present; (2) false negative means the error of missing a signal where there is one. There is a tradeoff between the types of errors: reducing the false positive probability could increase the false negative probability. The two errors are mostly correlated, and a receiver operating characteristic (ROC) curve can be used to analyze the two error probabilities for different detection parameter settings like thresholds [79].

This hypothesis test can be expanded for detecting and differentiating multiple signals, but more complicated algorithms are needed [79]. In addition, better signal models are needed to include interference, multi-path, fading, and other dynamic wireless environ-

ments [13]. For example the multi-signal classification problem can be formulated as:

$$\begin{aligned} H_0 &: x = n \\ H_1 &: x = hs_1 + n \\ H_2 &: x = hs_2 + n \\ &\cdots \\ H_N &: x = hs_N + n \end{aligned} \tag{3.2}$$

Our cyclostationary feature detection method follows the above hypothesis test.

3.1.1 Sensor Architecture

The above section gives the hypothesis test of signal detection. However, any discussion of RF signal detection must consider the hardware used to acquire signal samples because its characteristics fundamentally limit the detection performance.

A typical wireless sensor architecture is shown in Figure 3.1 [43]. It consists of a sensing unit, a processing unit, a power unit, a transceiver, an application specific location system, and a mobilizer if the sensor needs to move from one location to another. The sensing unit for a RF signal sensor is usually an RF front end.

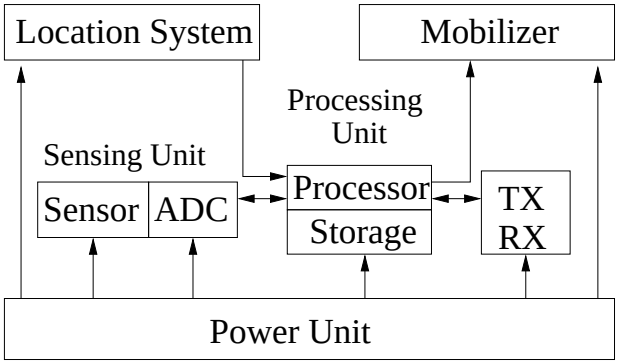


Figure 3.1: A typical sensor architecture.

3.1.2 Radio Front-end Architecture

In general, there are four types of RF architectures: tuned radio frequency receiver, super-heterodyne receiver, direct and low-IF conversion receiver. Depending on different applications, a sensor may use different RF front end hardware architecture. For example, direct conversion receivers are mostly used in wide-band receivers.

For signals within very and ultra high frequency range (30MHz to 3GHz) which is heavily used by industry and the public safety community, superheterodyne receivers are mostly used. It typical RF front end is shown in Figure 3.2. Five key performance parameters are [16]:

- sensitivity which defines the weakest signal that a receiver can detect and is usually determined by the various noise sources in the receiving system,
- selectivity which defines the ability of the receiver to detect the desired signal and reject others,
- spurious response which is a receiver's freedom from interference due to internally generated signals or their interaction with external signals,
- stability which is defined by the receiver gain and frequency change with temperature, time, voltage, etc.,
- dynamic range which is defined by the difference in power between the weakest signal that the receiver can detect and the strongest signal that can be supported (either in band or out of band) by the receiver without detrimental effects.

Details are available in [16] about the relationship between an RF front end's performance and the hardware components shown in Figure 3.2. However, the performance of any RF signal sensor is first determined by its hardware architecture components. For example, the noise figures of the RF filter and the LNA impact the overall receiver performance in ways that cannot be overcome by subsequent components.

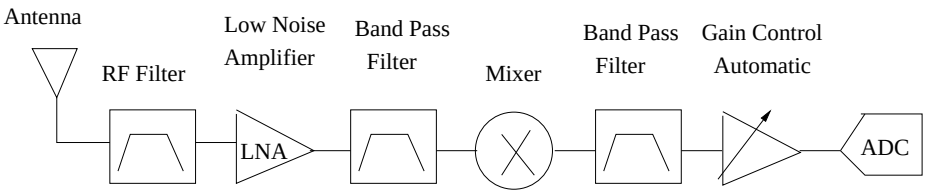


Figure 3.2: A IF conversion RF front end.

3.1.3 Signal Detection

Signal detection needs to determine whether a signal or multiple ones exist or not. However, a detector's performance can be quite different depending on the available amount of information about the signal. This section illustrates such a difference through two extreme cases. In one case, the detector has *a priori* knowledge of the signal so that it can

demodulate the signal; a match filter is used in the detector. In the other case, the signal's information is completely unknown, thus or so an energy detector is usually used.

Matched Filter

Given *a priori* knowledge of signals to be detected, a matched filter is the most optimal choice for signal detection since it maximizes received signal-to-noise (SNR) ratio [13]. Such knowledge includes modulation type and order, pulse shaping, packet format, etc. However, a dedicated receiver is needed to detect each type of signal and the receiver must perform timing and carrier synchronization, channel equalization, and even demodulation. But a matched filter, as a coherent detection method, only needs $O(1/SNR)$ samples to meet a given probability of detection constraint [80].

Energy Detector

As a non-coherent method which can be easily implemented, energy detection is extensively used in radiometry when a priori knowledge of signals is not available. Its typical implementation diagram is shown in Figure 3.3, where the processing gain is proportional to the Fast Fourier Transform (FFT) size N and averaging time T. Increasing N improves frequency resolution; extending averaging time reduces the noise power thus improves SNR. One detection result example is shown in Figure 3.4. It illustrates how an output above a threshold indicates the presence of a signal.

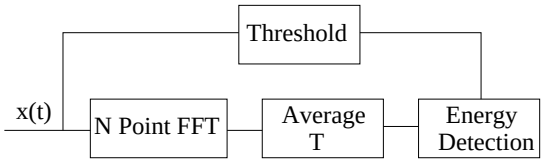


Figure 3.3: Implementation of an energy detector.

However, energy detection requires $O(1/SNR^2)$ samples to meet a probability of detection constraint [80]. Other drawbacks include: (1) The used threshold is highly susceptible to the radio environment. Even an adaptive method has difficulty when frequency selective fading exists. (2) An energy detector doesn't differentiate between modulated signals, noise and interference. (3) An energy detector doesn't work for direct sequence and frequency hopping signals which demand more sophisticated signal processing methods.

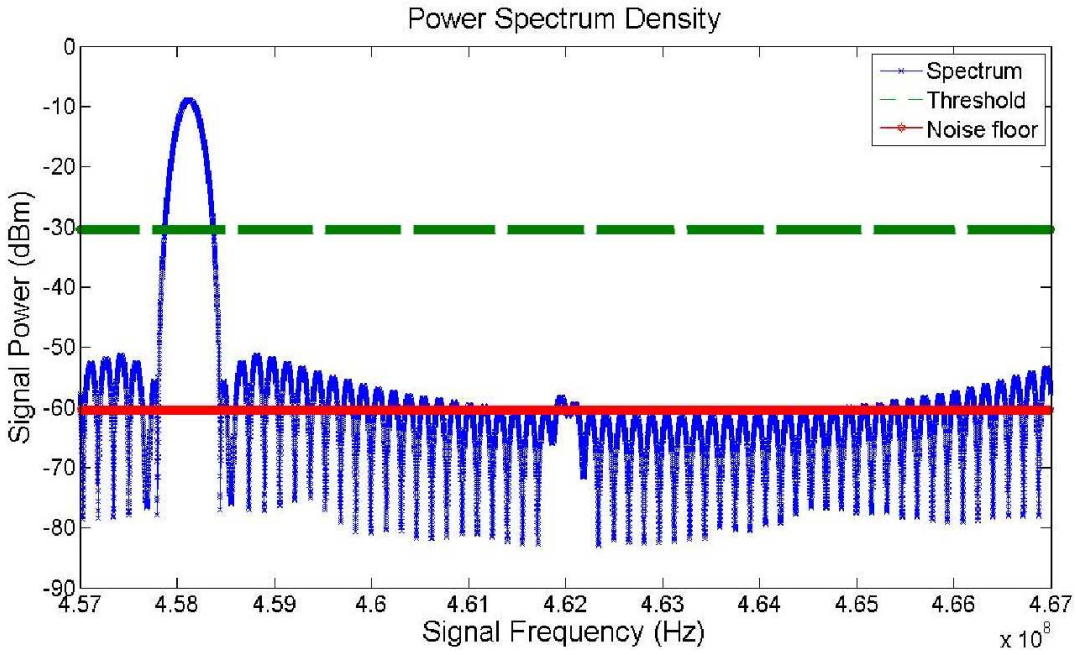


Figure 3.4: An energy detector result. ©2009 IEEE. Reprinted, with permission, from Ge et al., "A cooperative sensing based spectrum broker for dynamic spectrum access," IEEE Military Communications Conference (MILCOM), Boston, MA, 2009.

3.1.4 Likelihood Based Modulation Classification

If the signal types and their total number are known, modulation classification (MC) becomes a multiple composite hypothesis-testing problem, as formulated in Eq. 3.2. Likelihood (LB) based methods can used, where determining a signal is likelihood-ratio test (LRT). Further, Bayesian decision theory and maximum-likelihood (ML) estimation are usually used in such methods [79]. Three LB-MC methods were proposed in the literature: average LRT (ALRT), generalized LRT (GLRT), and hybrid LRT (HLRT) [81]. ALRT suffers from computational complexity and even mathematical intractability. Both GLRT and HLRT have implementation advantages and are applicable to different environments like Rician and Rayleigh fading. More details are available in [81]. In our prototype DSA network, nodes must differentiate signals transmitted by primary users from those transmitted by peer nodes. Therefore, signal recognition methods are needed. We used modulation classification, a popular method in signal recognition.

3.1.5 Feature Based Modulation Classification

Generally, a signal detector has a knowledge base of widely used signal types and their features within a frequency range, but not their exact number of them. Signal features in

the time and frequency domain can be used to classify signal types. Widely used signal features include:

- instantaneous amplitude, phase, and frequency,
- wavelet transform,
- cumulants,
- Higher order statistics, e.g., cyclostationarity.

Based on the above features, many machine learning and pattern classification algorithms [79] can be used for signal classification. Examples include hidden Markov models (HMM) [82], multi-layer neural networks [83], k-nearest neighbors algorithms (k-NN) [83], and support vector machines (SVM) [84].

3.1.6 Modulation Classification and Signal Synchronization

Beyond modulation classification, further extraction of signal features and parameters can find important applications for SDR and CR, for example, in signal interception. Those parameters include carrier frequency, symbol period, signal and noise power, equalization, etc.

The author's colleagues developed an advanced signal processing method called Universal Classifier and Synchronizer (UCS) algorithm [85]. UCS uses a radio signal's time and frequency domain features to classify and synchronize with a received signal and to provide all parameters needed for physical layer demodulation, without knowing any prior modulation information. It adopts a multi-step searching and decision method to identify signal parameters like carrier frequency, symbol timing, and modulation. Its theoretical analysis and detailed performance results are available in [85]. The current UCS is able to classify AM, FM, MPSK, QAM, MFSK and OFDM modulations based on over-the-air radio signal samples collected either by the GNU Radio/USRP or the Anristu Signature Signal Analyzer. The signal recognition method in our prototype DSA network applied UCS to differentiate signals transmitted by primary users from those transmitted by peer nodes.

3.2 Cyclostationary Feature Detector

Cyclostationary feature detection has advantages for spectrum sensing because of its ability to separate the SOI from noise and/or interference in the spectral correlation plane [86]. It

is well suited for signal detection and modulation recognition, signal parameter estimation, and the design of communication signals and systems. Unfortunately, the computational complexity of cyclic spectral analysis (which far exceeds that of conventional spectral analysis) limits its use as a signal and system analysis tool. One way to attack the computational complexity is to use a parallel algorithm and implement it on parallel computers.

Cell BE [71], a specially designed single-chip multiprocessor with low-cost and high parallel computing ability, provides this opportunity. It operates on a shared, coherent memory supporting one Power Processor Element (PPE) acting as the controller for the eight Synergistic Processing Elements (SPEs) processors designed especially for computation-intensive tasks. Cell BE's peak performance for single precision calculation is larger than 200 GFlops [87] and it provides a software development kit for parallel computing. The author used a PlayStation 3 with six usable SPEs to implement the computationally efficient algorithm FFT Accumulation Method (FAM) in order to estimate the Spectral Correlation Function (SCF) [88]. Specifically, the author parallelized the FAM algorithm on four SPEs to execute the computation intensive part and used the PPE to coordinate SPEs. He also used the data level parallelism supported by Single Instruction, Multiple Data (SIMD) and Vector Multimedia Extension (VMX) instructions and other acceleration techniques like Direct Memory Access (DMA), loop unrolling, double-buffering, etc. [89]. This chapter compares the computational complexity of running an SCF algorithm for different signal bandwidths, both sequentially on a GPP, and in parallel on a Cell BE. Moreover, the algorithm's computational speedup on a Cell BE is compared to a GPP based version for different bandwidths of signals.

3.2.1 Cyclostationary Spectral Analysis

Modulated signals have built-in periodicity like pulse trains, repeating spreading, and hoping sequences, characterized as cyclostationary. This information can be used for detecting a random signal with a particular modulation type in the presence of background noise and other modulated signals. Cyclostationary signals exhibit correlation between widely separated spectral components due to the spectral redundancy caused by periodicity. A signal process $x(n)$ is said to be cyclostationary in a wide sense if its mean (averaged over a finite time duration) and autocorrelation are periodic with a period T_0, i.e.,

$$\begin{aligned} M_x(t + T_0) &= M_x(t) \\ R_x(t + T_0, \tau) &= R_x(t, \tau) \end{aligned} \tag{3.3}$$

for all t and τ. Therefore, by assuming that the Fourier series expansion of $R_x(t,\tau)$ converges to itself, we can write [90]:

$$R_x(t,\tau) = \sum_{n=-\infty}^{+\infty} R_x^{\frac{n}{T_0}}(\tau) e^{i2\pi\frac{n}{T_0}t} \tag{3.4}$$

where the Fourier coefficients

$$R_x^{\frac{n}{T_0}} = \frac{1}{T_0}\int_{-\frac{T_0}{2}}^{\frac{T_0}{2}} R_x(t,\tau) e^{-i2\pi\frac{n}{T_0}t} dt \tag{3.5}$$

are referred to as cyclic autocorrelation functions and the frequencies $\{\frac{n}{T_0}\}_{n\in Z}$ are called cycle frequencies. Let α represent cycle frequency when the spectral correlation function (SCF) is defined as:

$$S_x^{\alpha}(f) = \int_{-\infty}^{\infty} R_x^{\alpha}(\tau) e^{-i2\pi f\tau} d\tau \tag{3.6}$$

There are generally two methods to estimate the signal SCF: frequency smoothing and time smoothing [91]. Time smoothing algorithms are considered to be more computationally efficient for general cyclic spectral analysis [91]. Given the signal $x(n)$, all time smoothing algorithms are based on the time smoothed cyclic cross periodogram:

$$S_x^{\alpha}(n,f)_{\Delta t} = \frac{1}{T}\langle X_T(n, f+\alpha/2) X_T^*(n, f-\alpha/2)\rangle_{\Delta t} \tag{3.7}$$

where $X_T(n, f+\alpha/2)$, called a complex demodulator, is the spectral components of signal $x(n)$. * is the complex conjugate operator. The cyclic cross periodogram is calculated by using a data tapering window of length T seconds sliding over the data for a time span of Δt seconds. Mathematically, computation of the complex demodulators is expressed as

$$X_T(n,f) = \sum_{r=-\frac{N'}{2}}^{\frac{N'}{2}} a(r)x(n-r)e^{-i2\pi f(n-r)T_s} \tag{3.8}$$

where $a(r)$ is a data tapering window of length $T = N'T_s$ seconds, T_s is the sample interval, and f_s is the sampling frequency.

After the complex demodulator has been computed, it is correlated with its conjugate

over a time span of Δt seconds. The correlation operation is expressed as

$$S_x^{\alpha}(n,f)_{\Delta t} = \sum_{r=0}^{N-1} X_T(r,f_1) X_T^*(r,f_2) g(n-r) \tag{3.9}$$

where $g(n)$ is a data tapering window of width $\Delta t = NT_s$ seconds, and $f = (f_1 + f_2)/2$ and $\alpha = f_2 - f_1$.

It is shown in [92] that the time smoothed cyclic cross periodogram converges to the spectral correlation function in the limit, as $\Delta t \to \infty$ followed by $f \to 0$, if the time windows $a(n)$ and $g(n)$ are properly normalized. Therefore, if $\sum_n a^2(n) = \sum_n g^2(n) = 1$, we have

$$\lim_{\Delta f \to 0} \lim_{\Delta t \to \infty} S_x^{\alpha}(n,f)_{\Delta t} = S_x^{\alpha}(f) \tag{3.10}$$

3.2.2 FFT Accumulation Method (FAM) Algorithm

The signal SCF can be estimated by Eq. (3.7) by two stages: first the complex demodulator of $x(n)$ is calculated through Eq. (3.8), then $X_T(n,f)$ is correlated over a time period of Δt seconds. The computational efficiency of this algorithm can be improved by decimating $X_T(n,f)$. For example, the improved algorithm only gets every other $L (L < N')$ sample for the second stage FFT. Therefore, Eq. (3.9) is modified to

$$S_x^{\alpha}(nL,f)_{\Delta t} = \sum_{r=0}^{\frac{N}{L}-1} X_T(rL,f_1) X_T^*(rL,f_2) g(n-r) \tag{3.11}$$

On the other hand, considering the frequency shifting for the second stage of SCF estimation, for example, from α to $\alpha + \varepsilon$, Eq. (3.9) becomes

$$S_x^{\alpha+\varepsilon}(n,f)_{\Delta t} = \sum_{r=0}^{N-1} X_T(r,f_1) X_T^*(r,f_2) g(n-r) e^{-i2\pi\varepsilon r T_s} \tag{3.12}$$

FAM is a computationally efficient algorithm to combine Eq. (3.11) and Eq. (3.12). It further quantizes ε into $\varepsilon = q\alpha$ [91]. Therefore, the demodulate correlation stage can be implemented in Fast Fourier Transform (FFT) too. The discrete S value $S_x^{\alpha_i + q\Delta\alpha}(nL, f_i)_{\Delta t}$ at the point with the cyclic frequency of $\alpha_i + q\Delta\alpha$ and the signal frequency of f_i in the

bi-frequency plane is shown in Eq. (3.13)

$$S_x^{\alpha_i+q\Delta\alpha}(nL, f_i)_{\Delta t} = \sum_{r=0}^{P-1} X_T(rL, f_k) X_T^*(rL, f_l) \\ g(n-r)e^{-\frac{i2\pi rq}{P}} \tag{3.13}$$

where α_i represents discrete cyclic frequency channels, $\Delta\alpha$ indicates cyclic frequency resolution in each channel and q is an integer; together $\alpha_i + q\Delta\alpha$ represents a discrete cyclic frequency value. f_k and f_l are the frequency values in correlating two spectral components; they decide the value of α_i by $\alpha_i = f_k - f_l$. Furthermore, L is a decimation factor for channelization in the frequency domain, P equals $\frac{N-N'}{L}+1$ where N is the total number of samples and N' is the number of samples used to calculate each complex demodulator $X_T(rL, f_k)$. The choice of N' must take into consideration that the time-frequency resolution product $\frac{N}{N'}$ must satisfy $\frac{N}{N'} \gg 1$ for a statistically reliable measurement [91] and that N' must be large enough to obtain the desired frequency resolution. L is usually chosen to be less than or equal to $\frac{N'}{4}$ [93]. More details about Eq. (3.13) and its variables' meaning are available in [91, 93, 94].

Essentially, the FAM algorithm can be described by Figure 3.5: the input signal is formed as an array with rows which are N' points long, with each succeeding row's starting point offset from the previous row starting position by L samples in the original sample sequence (input channelization). A window is applied across each row which is then Fast Fourier transformed and down converted to the baseband. The resulting array's columns represent constant frequencies, which are point-wise multiplied with the conjugate of other columns; the final stage is another round of FFT.

In [93], FAM is implemented in C on the general purpose processor. A Matlab version is also given by [95]. Following the above algorithm, the computation cost for a signal sequence with N samples is given by Eq. (3.14). Here, the first item represents the windowing, downconversion, and column multiplication computational cost, and the second item represents the two stages of FFTs computational cost which dominates the total value.

$$\text{Computational cost} = (3 + 2{*}N'){*}\frac{NN'}{L} + \frac{NN'}{L}(\log_2 N' + N'\log_2(\frac{N}{L})) \tag{3.14}$$

Considering the above parameter selection and the Nyquist Sampling Theorem, the computational requirement to cover a frequency range with 10 MHz bandwidth within one second is on the order of 1 Giga Floating point operations per second (GFlops) for a frequency resolution of 50Hz [94]. The computational cost is too high for current general

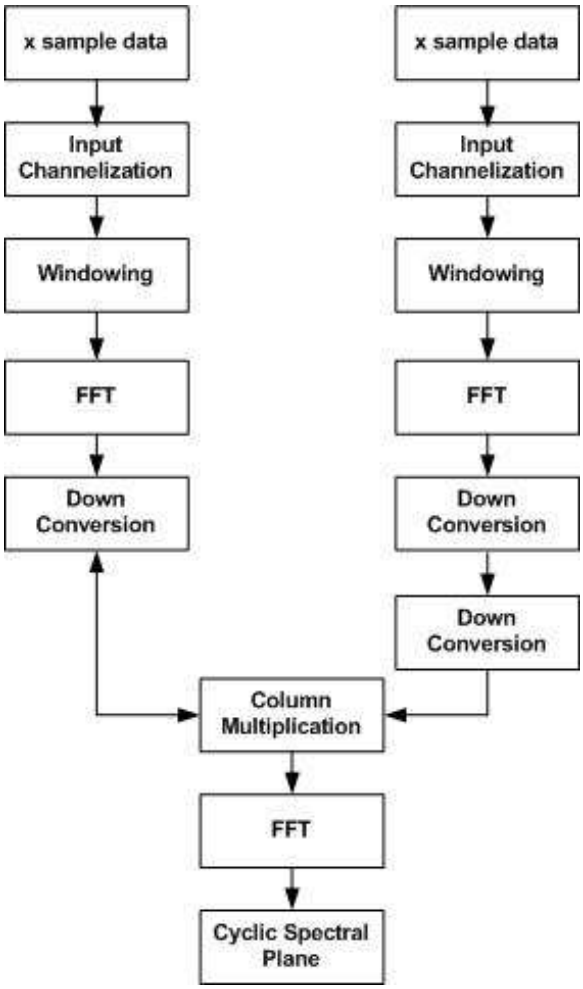


Figure 3.5: Sequential implementation of FAM algorithm. ©2008 IEEE. Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

purpose processors (GPPs). To demonstrate the computational requirement, the author ran the FAM algorithm for a different number of samples on an Intel Duo Core processor with 2.40 GHz CPU and 2 GB memory. Correspondingly, the amount of samples can represent different frequency resolutions given different parameter settings, as shown in Table 3.1, which also contains the time to accomplish the computation.

3.2.3 Parallel FAM algorithm on the Cell BE

Although cyclostationary feature analysis has the advantage of being able to differentiate modulated signals from noise, a key issue is the computational complexity associated with

Table 3.1: Serial FAM Computing Time. Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

N (samples)	N'	L	P	Resolution (Hz)	Time (s)
2^{18}	2^8	2^6	2^{12}	2^6	60.0
2^{17}	2^8	2^6	2^{11}	2^7	30.55
2^{16}	2^8	2^6	2^{10}	2^8	13.28
2^{15}	2^7	2^5	2^{10}	2^9	3.39
2^{14}	2^6	2^4	2^{10}	2^{10}	0.896
2^{13}	2^5	2^3	2^{10}	2^{11}	0.273

the calculation of a complete spectral correlation density function [88, 96]. This calculation is the main computing challenge in signal detection methods based on cyclostationary features. Even the computationally efficient algorithm FAM requires several seconds on a fast GPP, as shown in Section 3.2.2 and Table 3.1. Just based on such signal detection, it is not feasible for DSA implementations. For example, DARPA's XG system requires 0.5 second switching upon a primary signal's appearance [77]. Parallel computation can be used to attack this problem and two parallel computation structures are proposed for the Digital Frequency Smoothing Method in [88]. In addition, GPPs are not optimized for scientific computation since they are designed to run multiple general purpose tasks by using an operating system and a memory hierarchy [48].

The author designed a parallel FAM algorithm on a PlayStation 3 powered by the specially designed single-chip multiprocessor Cell BE with low-cost and high parallel computing ability [89, 71]. Cell BE uses a conventional high performance PowerPC core, also called the PPE, that controls eight simple SIMD cores known as SPEs. These SPEs are designed for computation- intensive tasks. To fully utilize Cell BE's capacity, it is required to distribute efficiently and effectively any computational task over the usable SPEs. This should take into account the Cell BE communication mechanism and memory bandwidth, SPEs' local memory size, SIMD operation, and computational models, etc. [89]. In this signal sensing approach, the FAM algorithm was parallelized to utilize four usable SPEs and the one PPE in a PlayStation 3. As shown in Eq. (3.14), the dominating computation cost lies in two stages of FFTs, which are also our focus for parallelization. The first stage has approximately N/L FFTs with the size of N', where the second stage has approximately N'^2 FFTs with the size of L. As mentioned in Section 3.2.2, we have $\frac{N}{N'} \gg 1$, $N' \gg 1$ and $L \leq \frac{N'}{4}$. Therefore, the FFT size in the first stage is moderate, while the

FFT size in the second stage is slightly larger. To estimate both sizes, we use an example. To cover 4MHz bandwidth (BW) with the resolution of 40Hz, we can get a data sample rate of 10Msps to have some frequency margin, N is 262144, and we can have $N' = 256, L = 64$,and $\frac{N}{L} = 4096$. Considering the SPE's local store size (256 KB), the above two FFT sizes can usually be accommodated by one SPE if the covered frequency BW is within tens of MHz. Therefore, this algorithm distributes FFTs equally among 4 usable SPEs and the PPE. To reduce the communication cost, the author chose a slightly larger N' so that each SPE could receive larger continuous data chunks each time, and the total number of parallel tasks was also reduced. The data independence among FFTs in either stage also enabled the parallel mechanism.

Specifically, as shown in Figure 3.6, the FAM used in [93] is parallelized. The input channelization is processed on the PPE, then $\frac{N-N'}{L} + 1$ FFTs operations are parallelized on 4 SPEs which also runs windowing and down conversion before and after FFT operation respectively. Then the data are sent back to the PPE for column multiplications which are partitioned and executed by the second stage of FFTs on 4 SPEs. Finally the data are post-processed in the PPE. All the data exchange between the PPE and SPEs was done through double buffered Direct Memory Access (DMA) [89] which read 128 bytes of data for each DMA command and asynchronously execute a list of DMA transfers while the SPE operated on previously transferred data.

The above material discusses the parallelism granularity analysis at the task level supported by 4 SPEs and the PPE. Another level is the data level parallelism supported by SIMD and VMX instructions. Everything in the SPEs is done in quadword (16 byte) granularity. For single-precision floats this corresponds to 4 way SIMD operation. The author used the modified stride-by-1 FFT algorithm proposed in [97] so that the FFT array could be partitioned naturally, without data rearrangement, into vectors that can be executed in parallel by SIMD instructions. Other implementation details included loop unrolling, precomputed $\sin$ and $\cos$ arrays for FFT, double-buffering, using system-specific large TLB pages, etc. As a result, 0.446 second was needed to accomplish the parallel FAM on an 8 MHz BW signal at frequency resolution of 512 Hz (32768 samples) which was around 7.6 times faster on a serial FAM than on the previous GPP (3.39 seconds).

3.2.4 Noise and Multi-path Impact on Signals' SCF

Hidden Markov Model, Neural Network, and other pattern recognition algorithms are proposed to classify several modulated signals' second order cyclostationary features [98, 99]. However, second order cyclostationary features cannot differentiate among higher order

phase-shift keying and quadrature amplitude modulation besides binary phase-shift keying [90]. In addition, theoretically, cyclostationary feature detection has the potential to differentiate modulated signals and noise in low signal to noise ratios [86] because noise doesn't have this feature. However, only a finite length of signal duration is usually used for detection and this impacts the detection performance, as shown in [100]. Besides, multi-path fading also has an impact on cyclostationary feature analysis [101]. Therefore, this section will first analyze signals' numerical SCF, specifically using FAM algorithm for some finite number of samples, under different levels of SNR and multi-path environment. After that it discusses a signal detection method designed by the author based on SCF.

To analyze the distinct SCF features for different modulated signals in the numerical domain, first the author simulated band-limited signals with different modulations and using root raised cosine pulse shaping (32 order coefficient). Then the author added AWGN and multi-path fading. A QPSK signal example is shown in Figure 3.7 which uses 16384 Hz sampling frequency and 1024 Hz carrier frequency.

Second, the author ran the FAM algorithm on different digital signals: M-PSK, M-FSK, M-QAM, and OFDM and find their SCF shape and distribution in the bi-frequency plane, as shown in Figure 3.20. All the SCF values shown in in this chapter calculated from the FAM algorithm are normalized magnitude values. All the modulated baseband signals are pulse shaped by a root raised cosine filter before they are up converted to a carrier frequency. All the figures in this chapter use 64 for N', 1024 for P, and 8 for L in the FAM algorithm.

As we can see the SCF is characterized by a combination of different shapes and locations in the bi-frequency plane. Specifically, it contains a sequence of 2D planes which are different for different modulation types.

Third, the author analyzed the numerical SCF of pure white Gaussian noise. He ran the FAM algorithm on white Gaussian noise at power levels from -30 dB to 20 dB relative to a unit power signal and found the SCF distributions were almost all the same as shown in Figure 3.8 at $\alpha \neq 0$ where the SCF is equally distributed on the bi-frequency plane and with normalized value of 0.2. While the SCF at $\alpha = 0$ is flat and the average value increases as power level decreases.

Fourth, BPSK was used as an example to analyze its SCF at different noise levels, as shown in Figure 3.9, 3.10, 3.11, and 3.12. As we can see, as the SNR decreases, the SCF pattern changes.

Fifth, the author also simulated Rayleigh multi-path fading using the model from [102]. He assumed that the delay power profile and the Doppler spectrum of the channel were separable and the multi-path fading channel was therefore modeled as a linear finite impulse-

response (FIR) filter [102]. One multi-path example is shown in Figure 3.13 and it impacts SCF pattern too. Three examples, BPSK, 8QAM, and OFDM based on QPSK signals are shown in Figure 3.14, 3.15, and 3.16 respectively.

3.2.5 SCF Signal Detection Algorithm

Although different modulated signals have distinct patterns, the difference focuses on very small specific locations in the bi-frequency plane among some modulations like QPSK, 8PSK, and QAM. Most importantly, noise and multi-path together dramatically impact the SCF pattern as shown in Figure 3.9, 3.10, 3.11, 3.12, 3.13, 3.14, 3.15, and 3.16, which makes signal classification very challenging. The author only used cyclostationary features to detect signals' existence even though different modulated signals' cyclostationary features were used. Several detection algorithms are proposed in [103] based on spectral correlation theory.

We can decouple the SCF features in the bi-frequency plane: one part corresponds to the null cycle-frequency ($\alpha = 0$) which actually represents the conventional power spectral density (PSD); the other part corresponds to the rest of the bi-frequency plane ($\alpha \neq 0$) [96]. In addition, since SCF is an even symmetric function of f and a Hermitian symmetric function of α [92]:

$$S_x^{\alpha}(-f) = S_x^{\alpha}(f)$$
$$S_x^{-\alpha}(f) = S_x^{\alpha}(f)^*$$

Therefore, we only need to focus on a quarter of the bi-frequency plane. In addition, as we notice in Figure 3.8, 3.9, 3.10, 3.11, and 3.12, the noise SCF is approximately equally distributed on the plane at $\alpha \neq 0$ and its value is less than signal SCF. Therefore, we can average the SCF over the entire bi-frequency plane and subtract it out of the original SCF in the first part of the detection algorithm. Next, signal SCF features consist of a series of 2D planes at discrete α values and each plane is symmetrical around a peak. Therefore, we can integrate local SCF over signal frequency on such planes and average the result using the algorithm in Eq 3.15 (where α is constant for each plane).

$$y_{\alpha}(f) = \frac{1}{\Delta f} \int_{f-\Delta f/2}^{f+\Delta f/2} (S_x^{\alpha}(\nu) - \bar{S}_x^{\alpha}(\nu)) d\nu \tag{3.15}$$

where $y_{\alpha}(f)$ is a local signal SCF strength indicator in the bi-frequency plane. $\bar{S}_x^{\alpha}(\nu)$ is the averaged SCF value over each plane (α is constant). The integral range is decided by the

following constraints:

$$(S_x^\alpha(\nu) - \bar{S}_x^\alpha(\nu)) > 0 \, over \, (f - \Delta f/2) \leq \nu \leq (f + \Delta f/2)$$
$$S_x^\alpha(\nu) \text{ is locally maximum at } \nu = f$$

y_α can be used to detect signal's presence since it indicates local signal SCF strength. An example of y_α with $f = 0$ for QPSK signal at the carrier frequency of 2^{10} Hz and the SNR level of -20dB is shown in Figure 3.8.

As shown in [92], Figure 3.8, 3.9, 3.10, 3.11, and 3.12, SCF global maximum peaks happen at $(\alpha = 0, f = \pm f_c)$ and $(\alpha = \pm 2f, f = 0)$ for M-ary PSK, QAM, and continuous FSK signals (f_c is the carrier frequency). A series of local maximum peaks occurs at discrete locations decided by the carrier frequency f_c and discrete α decided by symbol rate. Therefore, we can just utilize the global maximum peaks to detect signals at $\alpha \neq 0$.

The hypothesis test at f_i in Eq 3.1 is therefore:

$$H_1 : if \; y_\alpha(f_i) \geq \delta \; and \; f_i \text{ is a local maximum point}$$
$$H_0 : \text{otherwise} \tag{3.16}$$

where δ is decided by the noise's average SCF in the bi-frequency plane, as shown in Figure 3.20.

For $\alpha = 0$, the signal detection is simply a power spectrum density (PSD) detection which uses either a fixed or adaptive threshold to detect signal presence [103]. The detection algorithm proposed here combines both detection results made at $\alpha \neq 0$ and $\alpha = 0$.

3.3 Experiment Simulation and Results

The simulation experiment assumed that the SOI was completely unknown including its frequency and modulation type. Since most signal detection methods first down-convert signals from the RF band into an RF band, the author simulated different digital signals including M-ary PSK, M-ary FSK, M-ary QAM, and OFDM at an IF band. Therefore, the proposed detection algorithm had to search the whole RF band to detect any signals' presence. In addition, all the baseband signals used root raised cosine pulse shaping before the up-conversion to the carrier frequency, which mimics pulse shaping in real communication systems. Then, AWGN was added to the signals at different SNR levels to simulate AWGN channel conditions, some examples are shown in Figure 3.20. Next, multi-path fading was added together with different SNR levels of AWGN to simulate both noisy and

Table 3.2: Signal detection performance on different signals at different SNR levels (averaged on 20 rounds of simulation on 8248 signal samples). Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

Signals	0dB	-5dB	-10dB	-15dB	-20dB
BPSK	100%	100%	100%	95%	15%
QPSK	100%	100%	100%	80%	10%
8PSK	100%	100%	100%	90%	10%
OQPSK	100%	100%	100%	90%	10%
MSK	100%	100%	100%	90%	10%
2FSK	100%	100%	100%	95%	10%
4FSK	100%	100%	100%	95%	5%
8QAM	100%	100%	100%	95%	10%
16QAM	100%	100%	100%	95%	10%
OFDM1	100%	100%	100%	80%	5%
OFDM2	100%	100%	100%	80%	5%

Table 3.3: Signal detection performance on different signals at different SNR levels with multi-path fading. Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

Signals	0dB	-5dB	-10dB	-15dB	-20dB
BPSK	100%	100%	100%	85%	10%
QPSK	100%	100%	100%	75%	5%
8PSK	100%	100%	100%	75%	5%
OQPSK	100%	100%	100%	75%	10%
MSK	100%	100%	100%	75%	10%
2FSK	100%	100%	100%	85%	5%
4FSK	100%	100%	100%	80%	5%
8QAM	100%	100%	100%	90%	10%
16QAM	100%	100%	100%	70%	5%
OFDM1	100%	100%	100%	90%	10%
OFDM2	100%	100%	100%	90%	5%

fading channels. The parameter settings for signal simulation were: the sampling frequency was 16384Hz, the RF carrier frequency was 1024Hz, the FSK's frequency separation was 100Hz, the OFDM block size was 16 and its cyclic prefix was 7, the pulse shaping filter roll-off was 0.25, and the filter order was 32.

To detect the signals, first the author ran the FAM algorithm to get their SCF value in the bi-frequency plane. Then, he calculated the SCF's magnitude and normalized it to the range $[0, 1]$. Therefore, the proposed signal detection algorithm is based on real positive values within 0-1. Next, the author used the detection algorithm shown in Eq 3.15 and estimated any signal's existence subject to Eq 3.16. In addition, he also identified the centers of dense clusters like those shown in Figure 3.17 and used the cluster centers to estimate the carrier frequencies. The author was able to do so because SCF global maximum peaks occur at $(\alpha = 0, f = \pm f_c)$ and $(\alpha = \pm 2f, f = 0)$ for M-ary PSK, QAM, and continuous FSK signals (f_c is the carrier frequency). This estimation was also confirmed in the plane of $\alpha = 0$ which was a PSD detection problem. The parameter settings for the FAM and signal detection algorithm were: $N' = 64$, $P = 1024$, and $L = 8$, $\delta = 0.05$.

For pure AWGN channel, the detection of signal existence for 20 rounds of simulation is shown in Table 3.2, where OFDM1 and OFDM2 represent OFDM based on BPSK and QPSK respectively. As we can see, the signal detection algorithm does not have any errors until SNR drops to -15dB, and it doesn't have any statistical meaning at -20dB SNR at current algorithm parameter settings.

Next, the author simulated both AWGN and Rayleigh multi-path fading channel. The parameter setting for Rayleigh multi-path fading simulation was: the sample time of the input signal $t_s = 1/16384$ s, the maximum Doppler shift $f_d = 100Hz$. They were for the frequency-flat ("single path") Rayleigh fading channel simulation parameters. For frequency-selective ("multiple path") fading channel, the author used six paths with the vector of path delays of [0, 1.2, 2.3, 6.2, 11.3, 15.4] $* t_s$ and the corresponding path gains were linearly related to the path delay. The signal detection result for 20 rounds of frequency-selective fading simulation is shown in Table 3.3. As we can see, the detection performance is almost the same with the result only under AWGN channels except that the missing error is higher at -15dB SNR.

3.4 Conclusion and Discussion

This chapter first gives an overview of widely used methods in signal detection and classification methods. It then focuses on the special feature based signal detector using cyclostationary features. To reduce the computation requirements, the author uses parallel

computing in analysis running on a Cell Broadband Engine (Cell BE). Specifically, he parallelizes the FFT accumulation method (FAM) algorithm to calculate spectral correlation functions (SCF) on four available Synergistic Processing Elements in a PlayStation 3. The author analyzes the proposed algorithm's computational speedup on a Cell BE compared to a GPP based version.

The author analyzed the impact of AWGN and Rayleigh multi-path fading on signals' SCF features in the bi-frequency plane (with axes of cyclic-frequency and frequency). Next, the author designed a spectrum sensing algorithm which uses the distinct SCF pattern of each modulated signal to detect its existence. He ran this algorithm on simulated signals including M-ary Phase-shift keying (PSK), Frequency-shift keying (FSK), Quadrature amplitude modulation (QAM), and PSK based Orthogonal frequency-division multiplexing (OFDM) under conditions of multi-path fading and different levels of SNR. He gave numerical results for the detector's performance.

As discussed in [80], there exist SNR walls for feature detectors though the relative locations of the SNR walls differ for different algorithms. It is important to exploit the exact location for a specific feature detector such as the cyclostationary feature detection described here. The author's experiment verifies the existence of an SNR wall for cyclostationary feature detection. Besides, the result shows that multi-path has an impact.

The techniques presented here were not implemented in our prototype because the nodes and other available equipment lacked the computational power to implement them. Nevertheless, I think they will be used in future DSA networks and are thus a needed contribution to the literature.

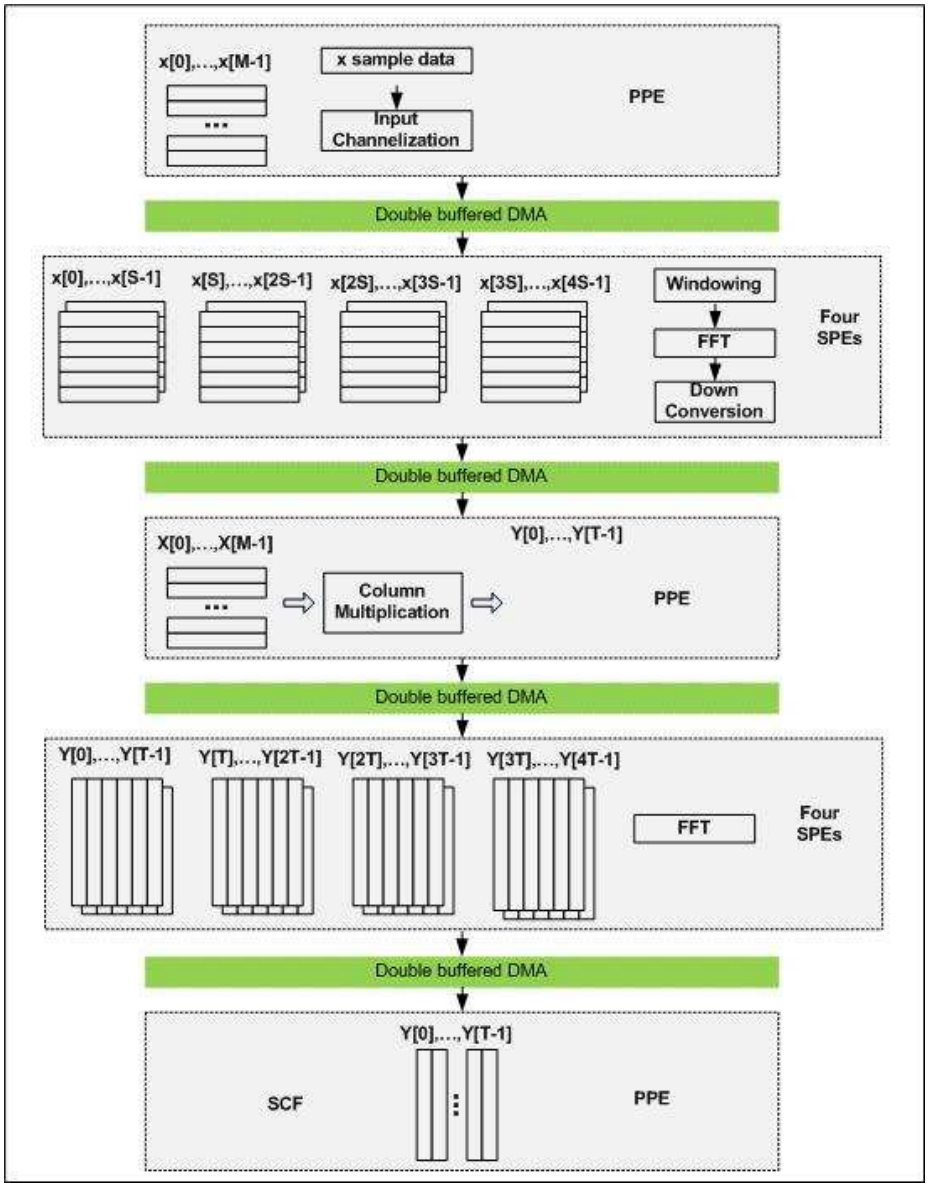


Figure 3.6: Parallelize FAM algorithm on the PlayStation 3. Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

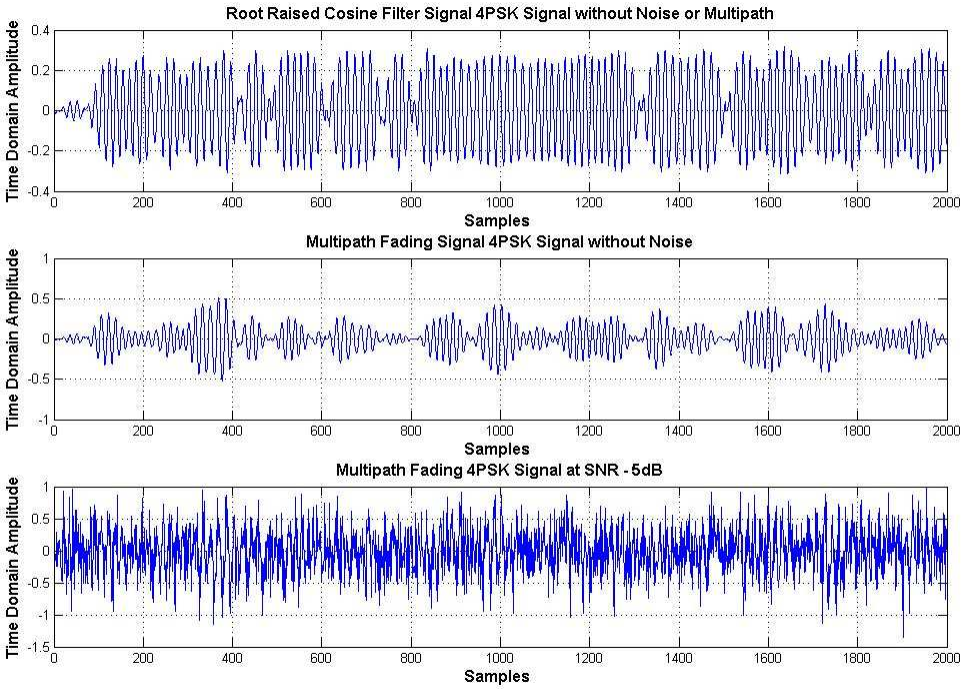


Figure 3.7: Simulated QPSK with pulse shaping, AWGN noise, and multi-path fading. Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

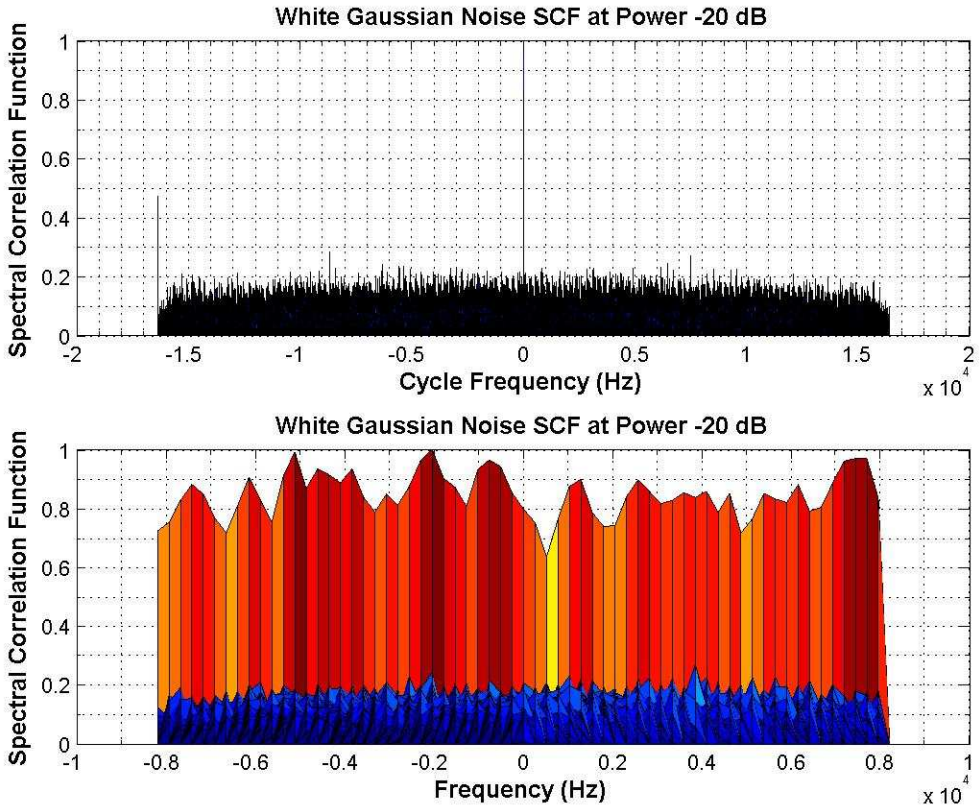


Figure 3.8: White Gaussian noise spectral correlation function at -20 dB power level. Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

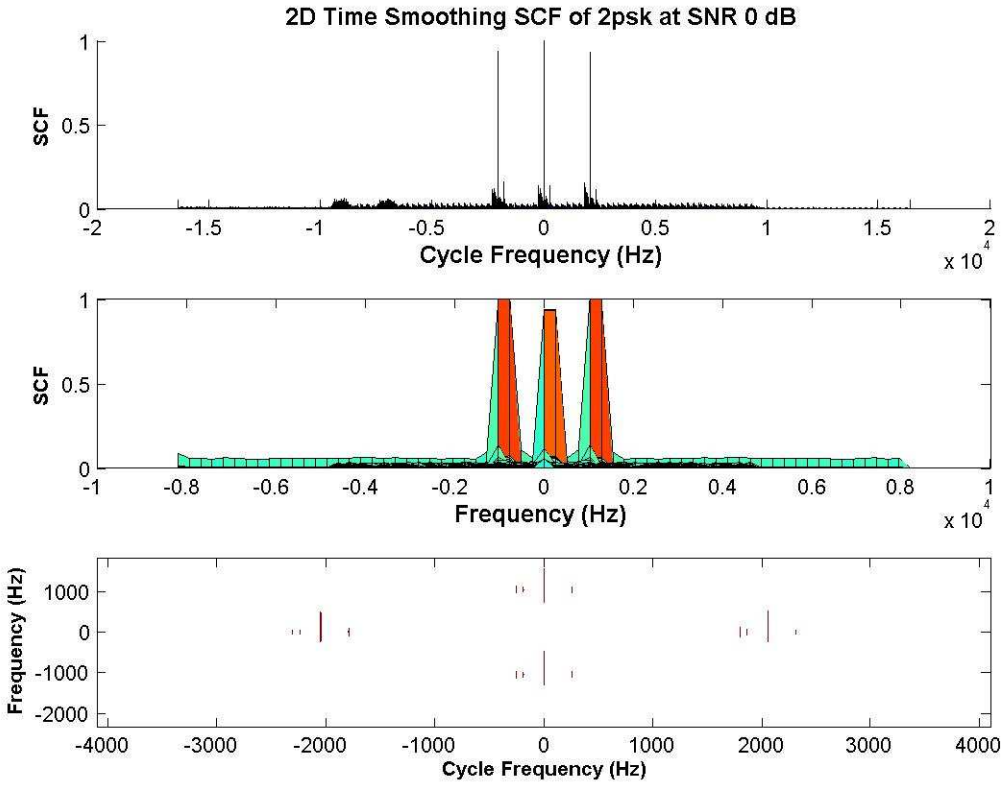


Figure 3.9: Bandlimited BPSK signal SCF at the SNR level of 0 dB. Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

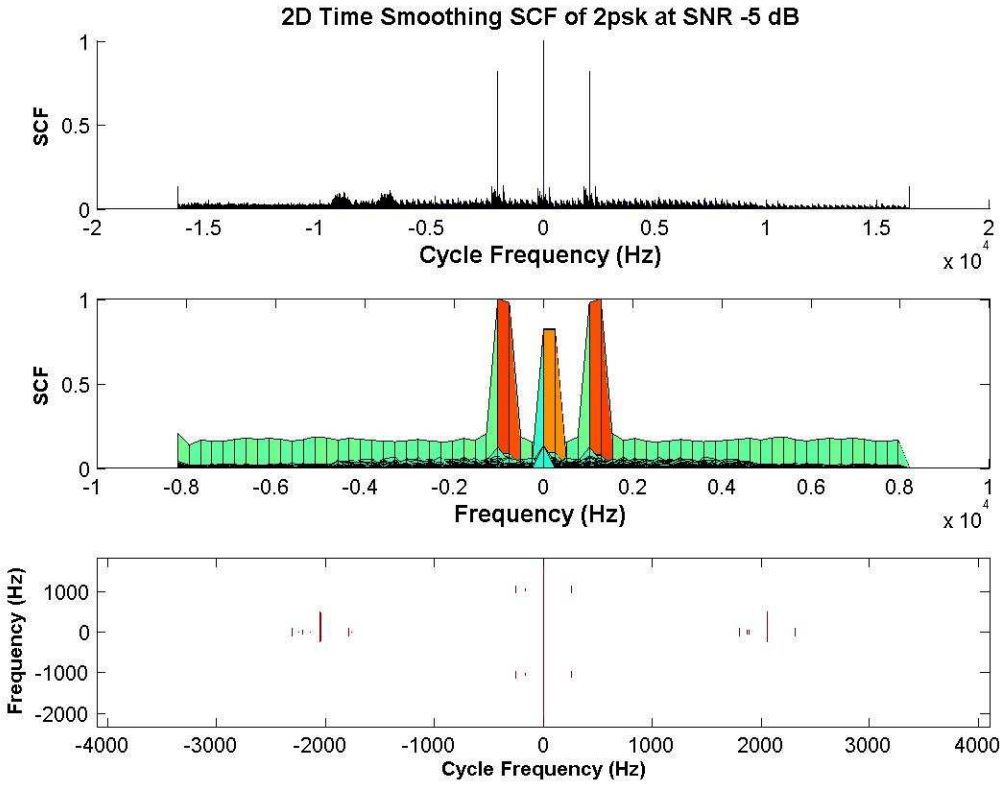


Figure 3.10: Bandlimited BPSK signal SCF at the SNR level of -5 dB. Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

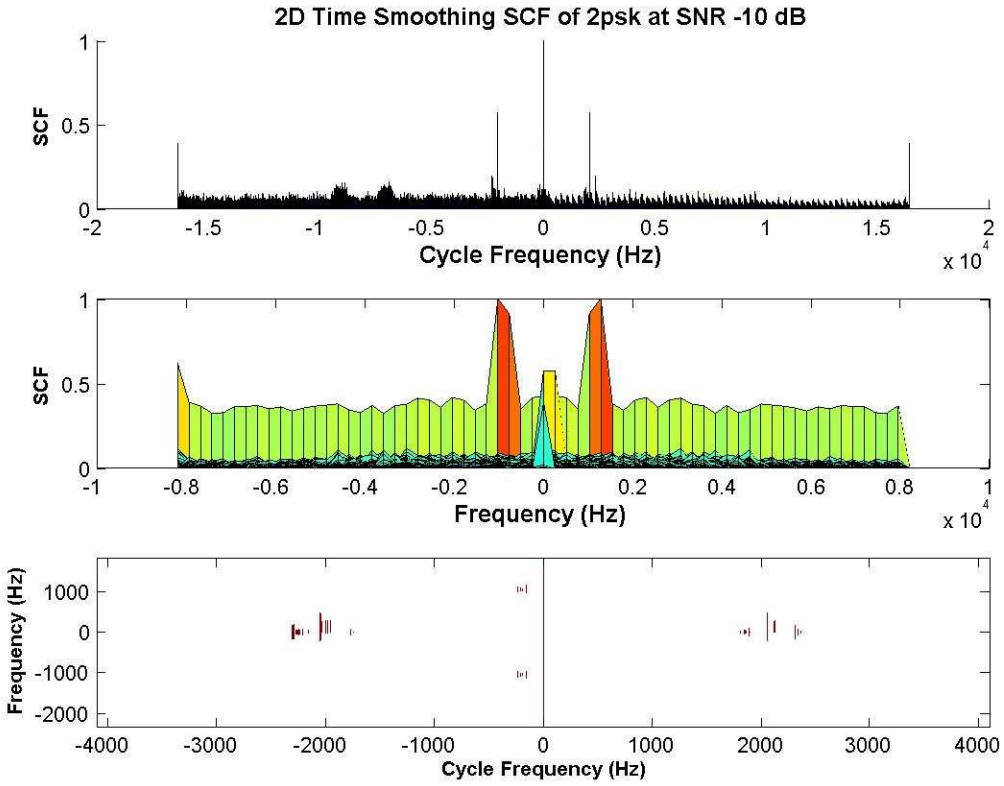


Figure 3.11: Bandlimited BPSK signal SCF at the SNR level of -10 dB. Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

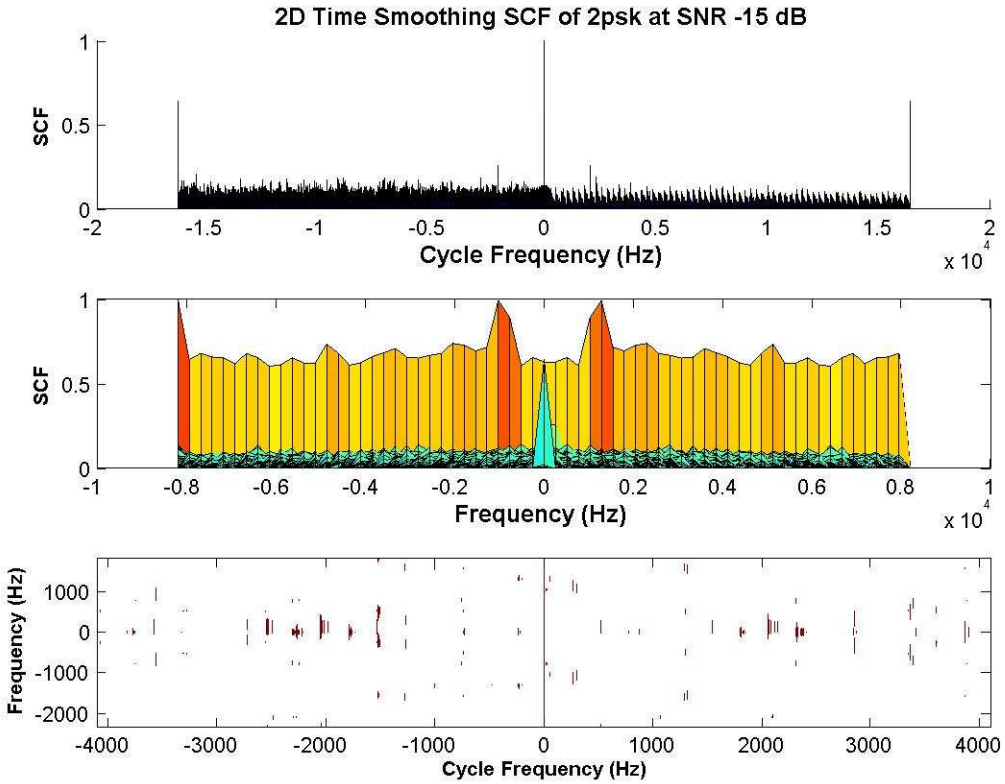


Figure 3.12: Bandlimited BPSK signal SCF at the SNR level of -15 dB. Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

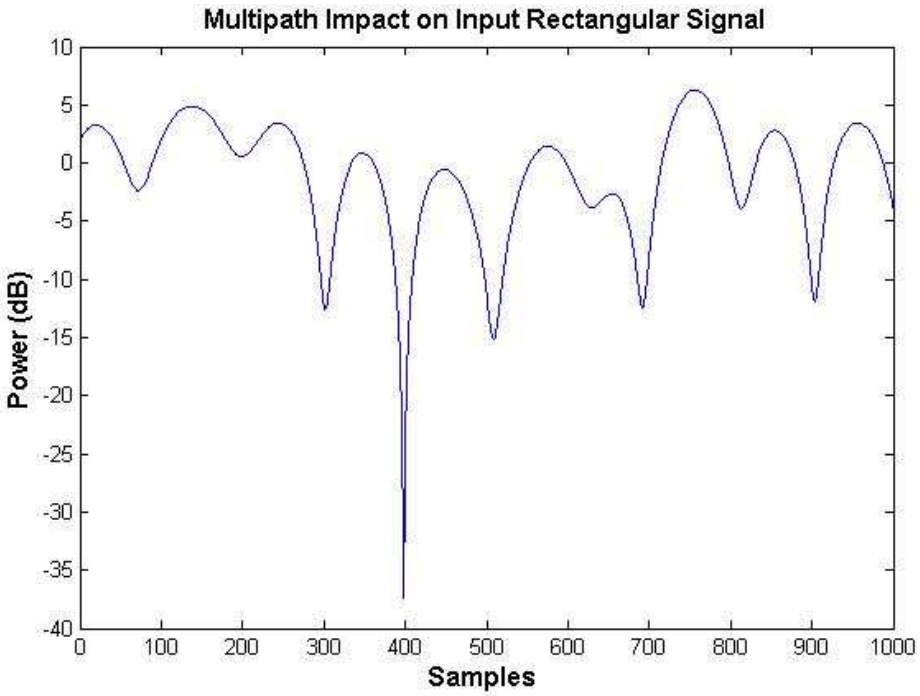


Figure 3.13: A multi-path filter response to a rectangular signal. Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

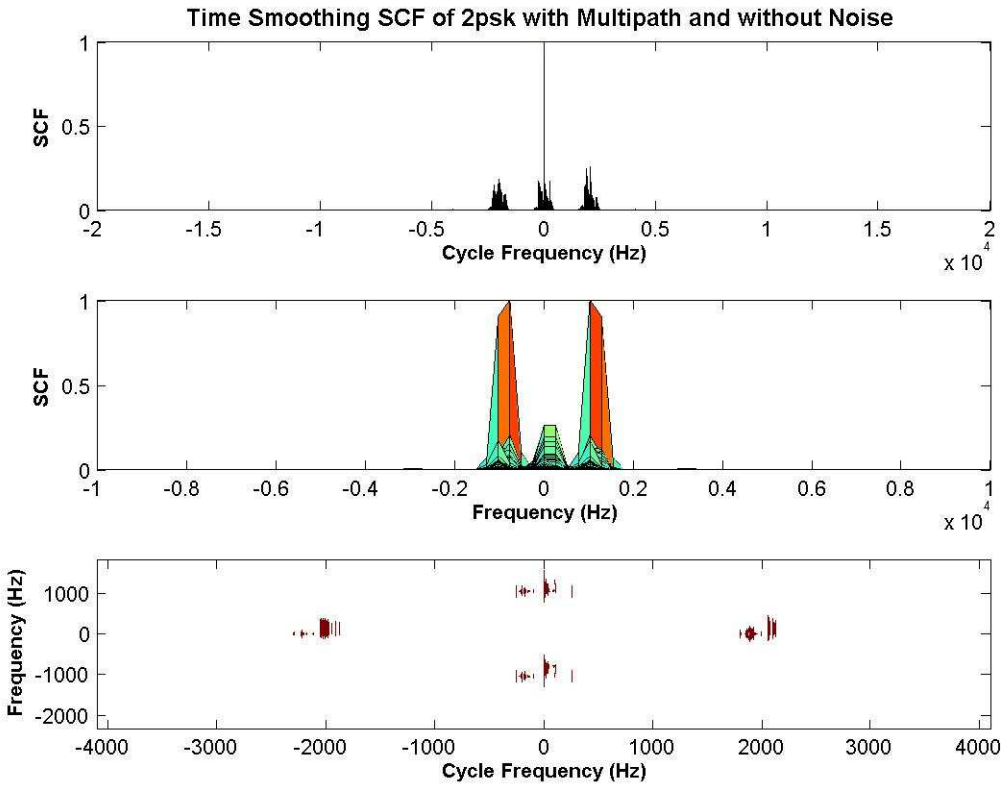


Figure 3.14: Multi-path impact on a BPSK signal's SCF. Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

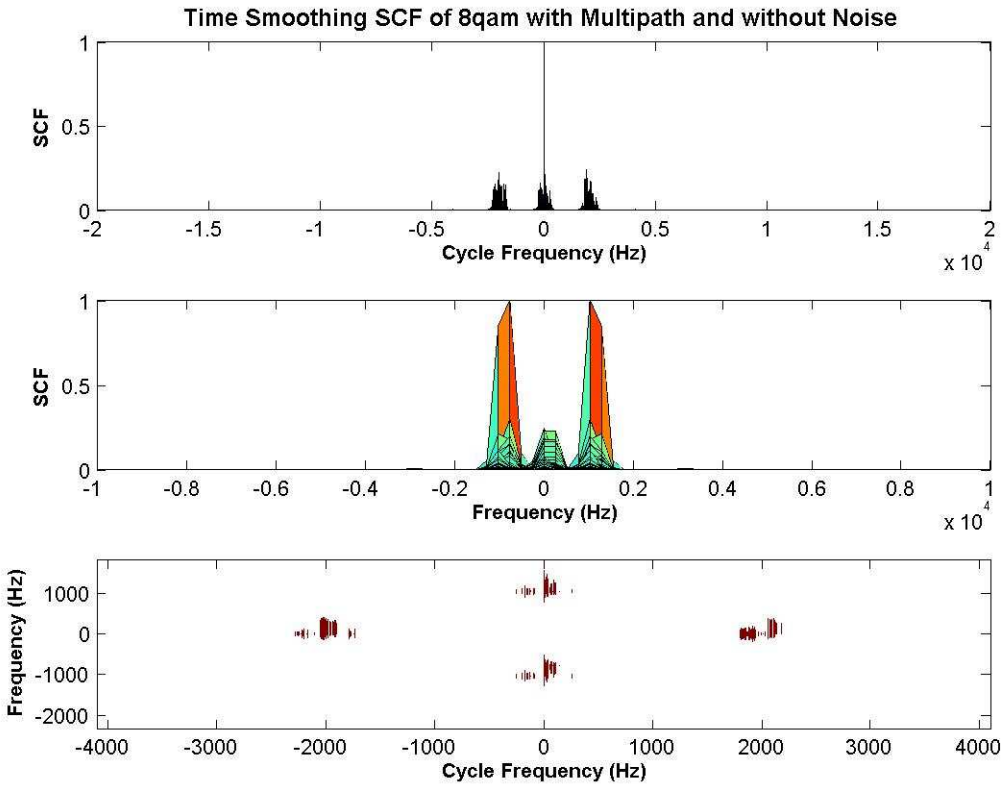


Figure 3.15: Multi-path impact on a 8QAM signal's SCF. Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

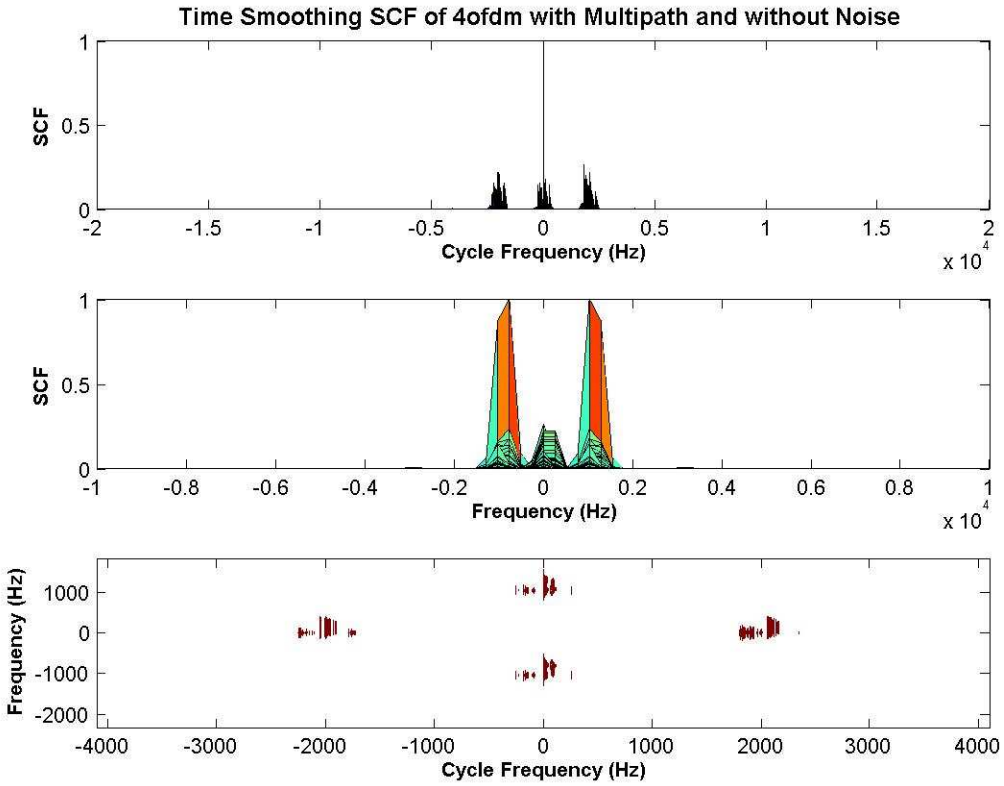


Figure 3.16: Multi-path impact on a QPSK based OFDM signal's SCF. Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

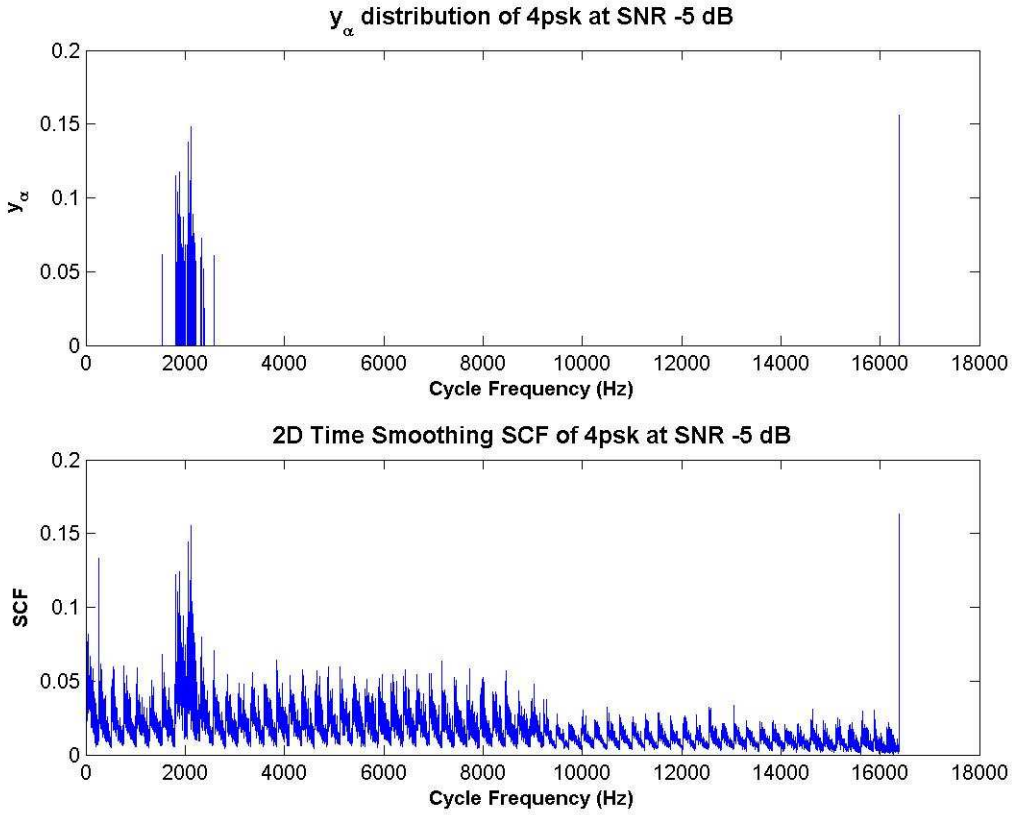


Figure 3.17: An example of y_α at $f = 0$. Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

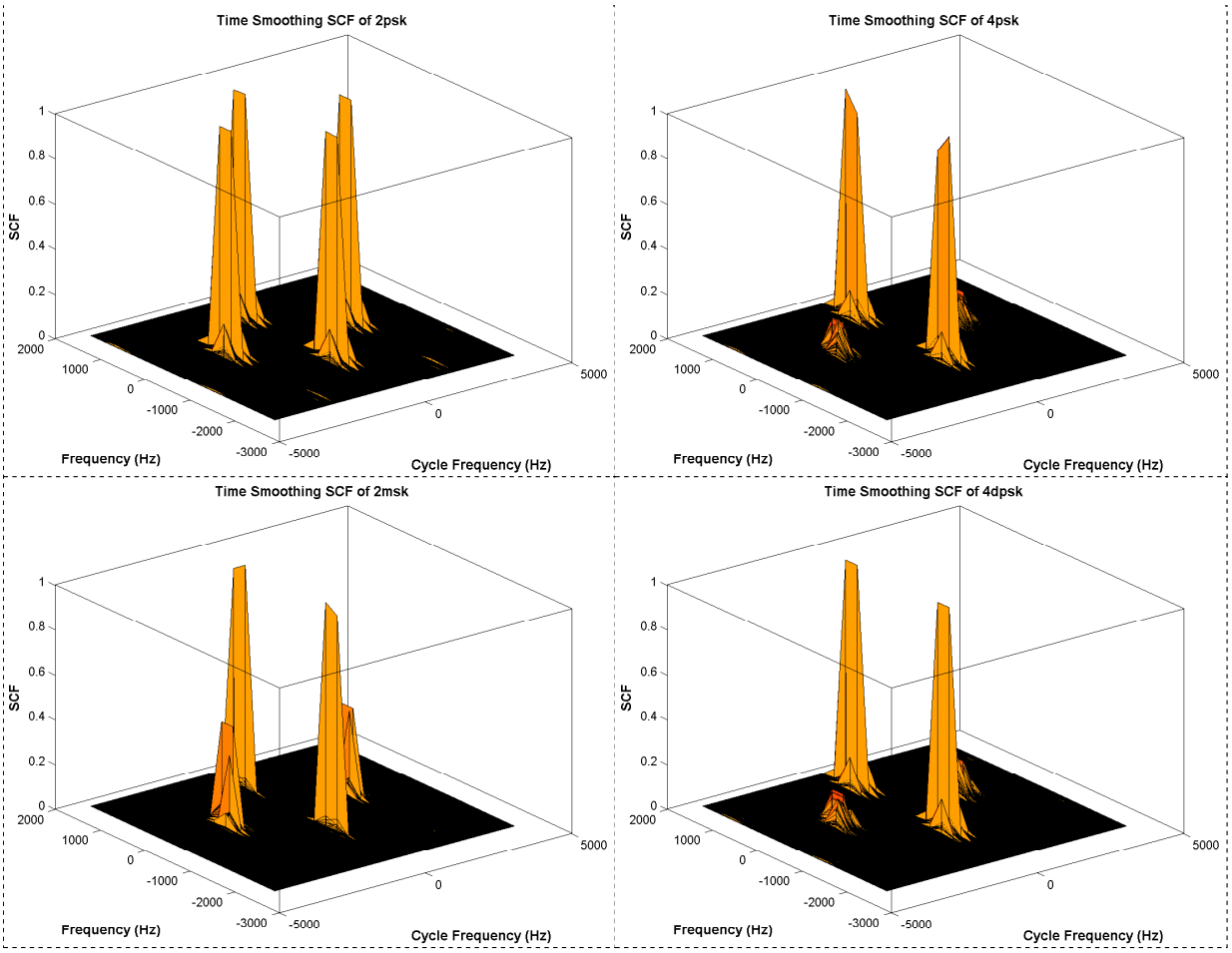


Figure 3.18: SCF of signals with modulations BPSK, QPSK, minimum-shift keying (MSK), and QDPSK (from left to right, up to bottom.) Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

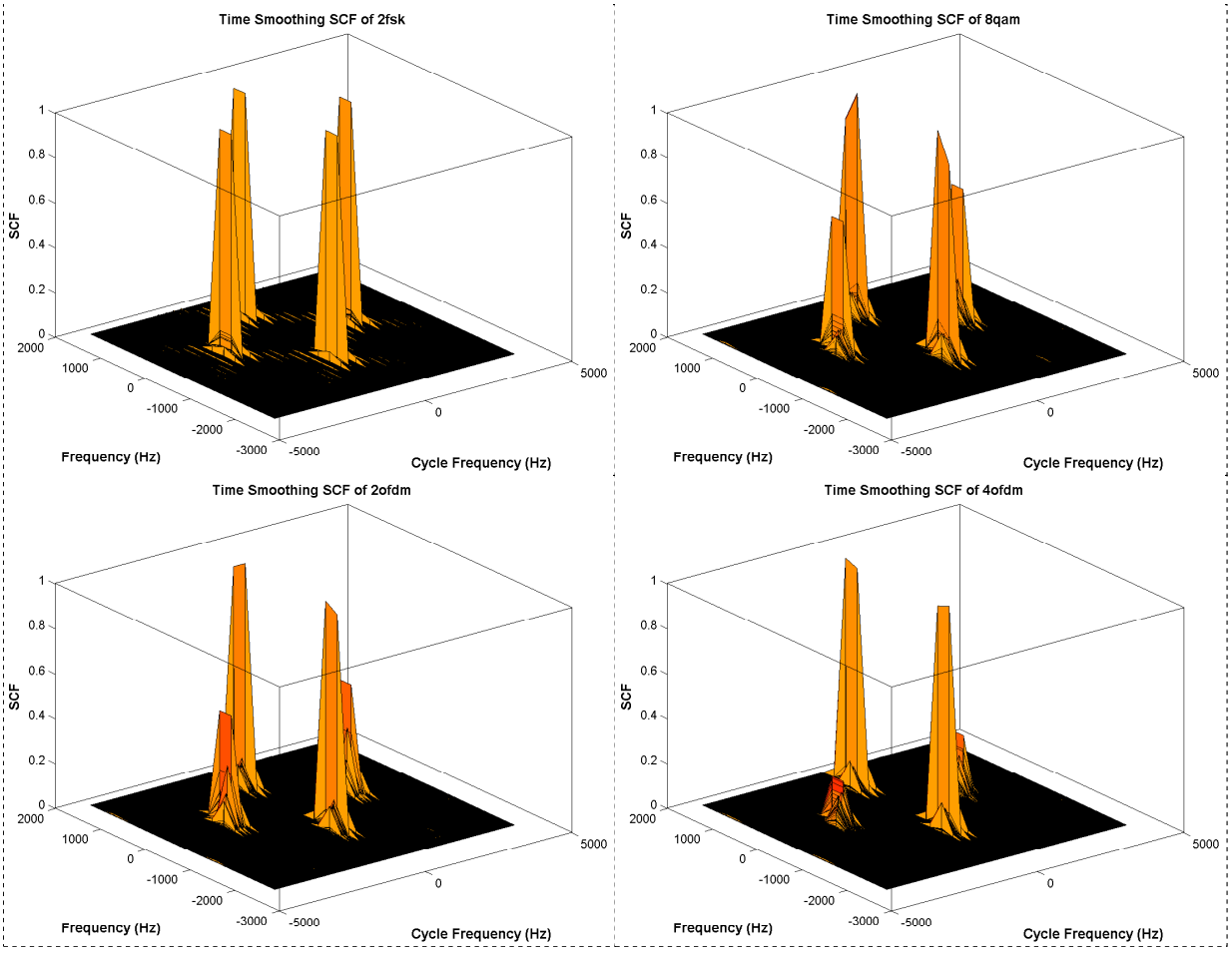


Figure 3.19: SCF of signals with modulations BFSK, 8QAM,OFDM with BPSK subcarrier, and OFDM with QPSK subcarrier (from left to right, up to bottom.) Reprinted, with permission, from Ge et al., “A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading,” The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

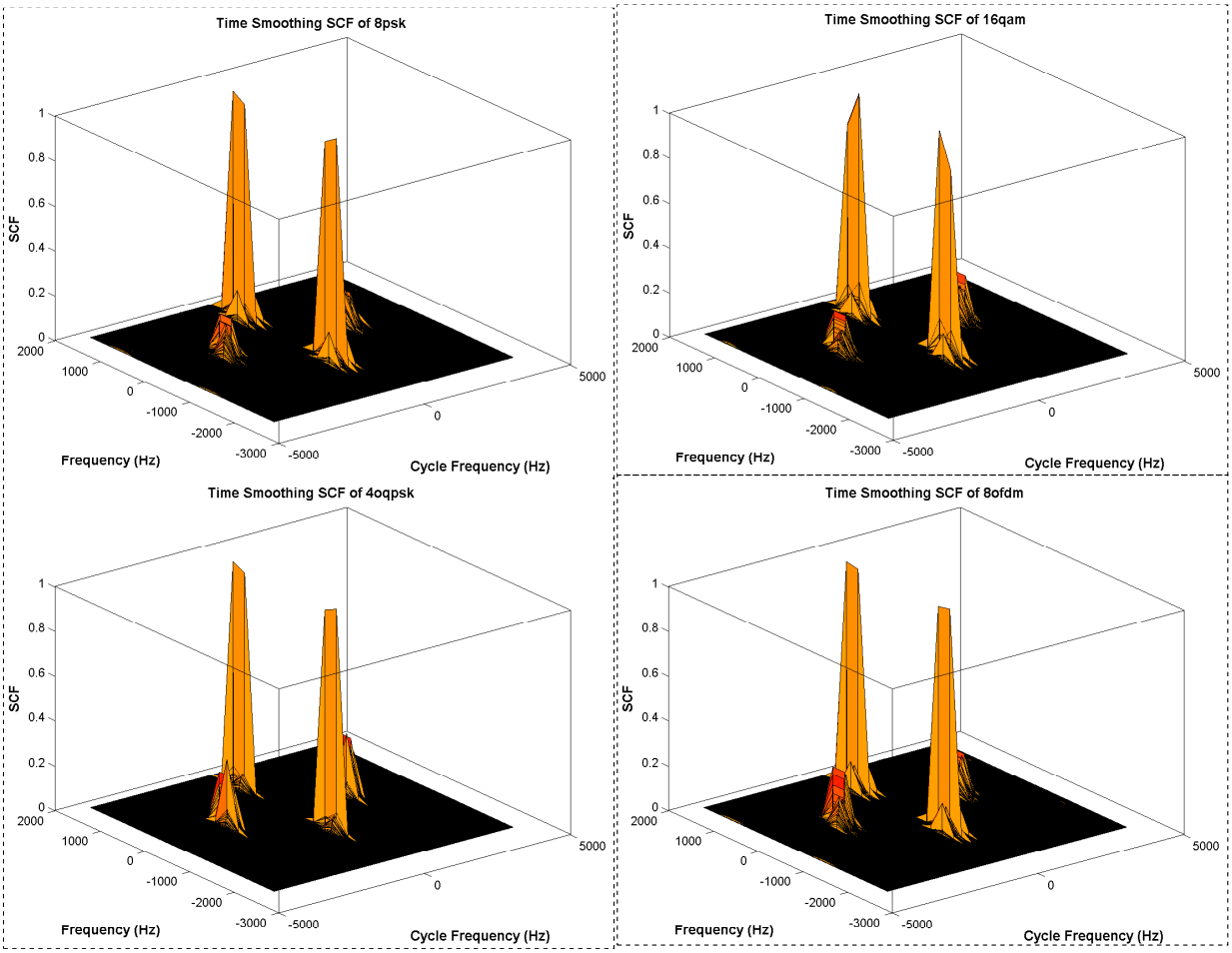


Figure 3.20: SCF of signals with modulations 8PSK, 16QAM, offset quadrature phase-shift keying (OQPSK), and OFDM with 8PSK subcarrier (from left to right, up to bottom.) Reprinted, with permission, from Ge et al., "A parallel computing based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading," The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN), Chicago, IL, 2008.

Chapter 4

Cooperative Spectrum Sensing for Dynamic Spectrum Access

Quickly and reliably detecting signals can be very challenging under dynamic and hostile conditions like shadowing, fading, interference, and multi-path [42, 44]. Because a single signal detector may fail under such conditions, cooperative spectrum sensing is widely endorsed [42, 104, 105]. Furthermore, as the radio environment varies, nodes in a DSA network must be aware of such variations and choose the right vacant channels to talk to each other. Essentially a radio environment map is needed [45]. To do so, a DSA network has to rely on cooperative spectrum sensing which usually disperses sensors geographically.

However, such a cooperative spectrum sensor network puts stringent conditions on both data latency and data reliability because secondary users should quickly and reliably be informed about incumbent signals' presence. Cooperative spectrum sensor data should be fused in a data fusion center. The result is needed by secondary users in managing spectrum sharing. This chapter designs spectrum data flow and process management systems for spectrum sensors and the data fusion center to reduce data latency and achieve high data reliability. It also describes a DSA broker that not only interfaces with cooperative spectrum sensors and fuses sensed spectrum data, but also manages secondary user's access to spectrum.

Cyclostationary feature based detectors can be sped up by using parallel computing, but this demands a long observation and processing time and, of course, multiple processors capable of running the algorithms. Therefore, the author, in his DSA network development uses other signal detection and classification methods developed by his colleagues and himself on different radio platforms. This chapter also introduces those sensors and their integration into a cooperative sensor network.

4.1 Introduction to Decentralized Detection

Decentralized detection is widely used in commercial and military surveillance systems, where multiple sensors (such as radars, sonics or infrared sensors, and cameras) are simultaneously employed to improve the overall system performance; for example, reliability or speed [106, 107, 108, 109]. This section briefly reviews some previous work in the field.

4.1.1 Distributed Detection

Tenney and Sandell [106], in their pioneering effort, have extended the classical Bayesian decision theory to the case of a distributed two-sensor network. The system they consider is shown in Figure 4.1, where each sensor observes the phenomenon (H) and perceives a variable (y_1 and y_2 respectively). The sensors carry out some local processing (γ_1 and γ_2 respectively) on the data (with results of u_1 and u_2 respectively) to reduce the communications bandwidth required between them and the fusion center. The fusion center, upon receiving the data from two sensors, makes a global decision (with the result of u_0) on what happens in the phenomenon. More specifically, Tenney and Sandell's paper considers a binary hypothesis testing problem, where H has a probability (P_1 and P_2) to be in either case H_1 or H_2, and each sensor sends only a single bit to the fusion center.

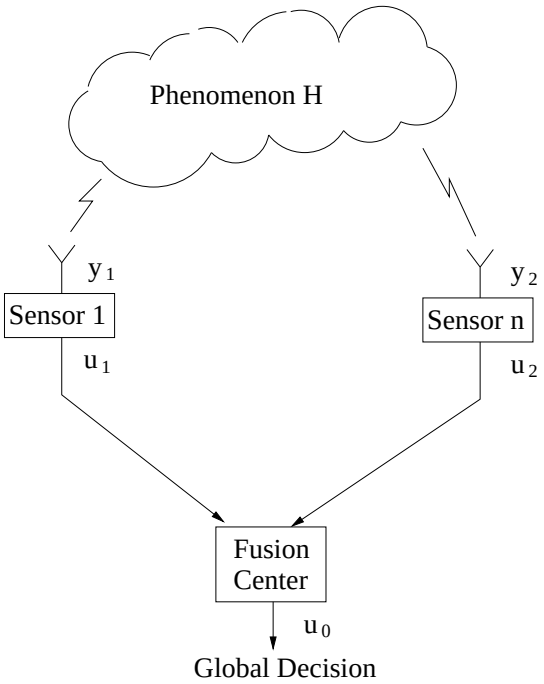


Figure 4.1: A distributed two sensor network.

The problem can be stated (where $i \in \{1, 2\}$):

$$\begin{aligned} H &\in 1, 2 \\ Pr(H = i) &= P_i \\ u_i &= \gamma_i(y_i) \end{aligned} \tag{4.1}$$

Separable cost functions can be defined

$$\begin{aligned} J(H, u_1, u_2) &= J_1(H, u_1) + J_2(H, u_2) \\ J_i(1, 2) &> J_i(1, 1) \\ J_i(2, 1) &> J_i(2, 2) \end{aligned} \tag{4.2}$$

The goal of the distributed system is to design $p(u_i|y_i)$ used to achieve a global optimal decision based on the above cost function comparison.

The results are surprisingly complex [106].

- The decision rule at each sensor is a likelihood ratio test. But the threshold depends on the decision rule at the other sensor and the conditional distribution of the measurement at the other sensor conditioned on the local measurement.
- Solutions to thresholds may be locally optimal but not globally so; two identical detectors may have different thresholds even for symmetric cost functions.

In the previous analysis, the fusion center decision rule is implicit. Chair and Varshney explicitly incorporate optimization of the fusion rule by weighting the incoming bits according to their reliability, which is a function of the local measurement model [107].

Tsitsiklis generalizes the problem to a multiple-sensor network, where each sensor transmits finite-valued messages to a fusion center that makes a final decision on one out of M alternative hypotheses [108]. Based on the binary hypothesis test setting in Eq.4.1 and 4.2, this problem is extended as a M-ary hypothesis test with D-ary alphabets as shown in

Figure 4.2 and below (where $i \in \{1, \ldots, N\}$):

$$
\begin{aligned}
H &\in 1, \ldots, M \\
N \text{ peripheral sensors, } D\text{-ary alphabet} & \\
\gamma_i &: y_i \rightarrow 1, \ldots, D \\
u_i &= \gamma_i(y_i) \\
\text{Data fusion center with local measurement} & \\
\gamma_0 &: y_1 \times u_1 \times \ldots \times u_N \rightarrow 1, \ldots, M
\end{aligned} \tag{4.3}
$$

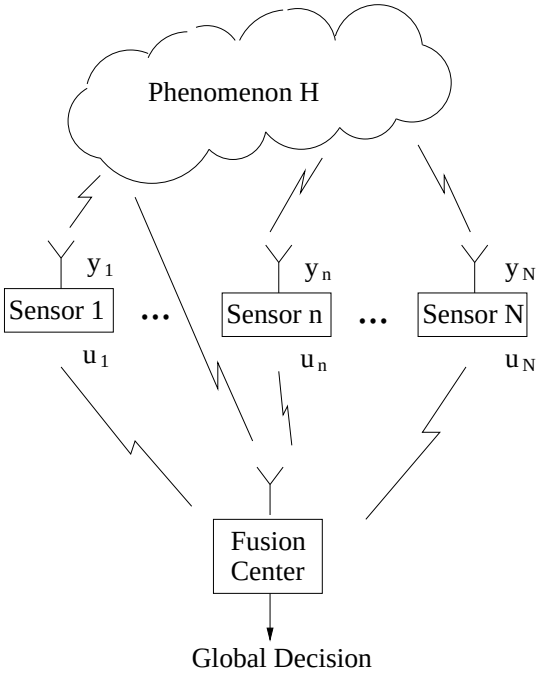


Figure 4.2: A distributed N sensor network.

Achieving minimum probability of error, as shown in [108] requires a search in the general case of all the decision rules, which is intractable. Approximation algorithms are suggested in [108].

4.1.2 Distributed Estimation

In distributed sensor networks, first each sensor sends to the fusion center of a summary of its own observations in the form of a message; the fusion center then makes a decision on the messages sent by all sensors. Both the data fusion center and the sensors carry out some amount of computation. The previous sections focuses on static sensor fusion with a star

topology, where the data fusion center has immediate access to local estimates from every sensors and makes only one decision based on all the data. However, the above analysis and approaches can be generalized to static sensor fusion with an arbitrary network topology at the expense of more time being required [110, 111].

Dynamic distributed estimation uses multiple sensors continuously to detect a random variable that is evolving in time or along space. It has applications in state estimation/tracking, identification, and random field estimation [112, 110]. This problem is shown in Figure 4.3, where N_s sensors continuously sense an evolving variable X. At each time step j, the variable is represented by X_j, while each sensor has a different observation (represented by z_j^i, where $i \in \{1, \ldots, N_s\}$). The goal of this sensor network is to estimate the variable at next time step based on each sensor's current measurement and their past knowledge, given that the probabilistic estimation of this variable based on the measurements at the initial time step $j = 0$ is known. Such a problem can be formulated from a Bayesian view as:

$$\begin{aligned} &\text{Given: } p(x_k|\{z\}_0^k) \\ &\text{Efficiently compute} \\ &p(x_{k+1}|\{z\}_0^k) \\ &p(x_{k+1}|z_{k+1}, \{z\}_0^k) \end{aligned} \tag{4.4}$$

where $\{z\}_0^k$ are the past measurements at all sensors . Two steps of iterative procedures are needed for prediction and update:

$$\begin{aligned} &\underline{\text{Prediction}} \text{ Chapman-Kolmogorov} \\ &p(x_{k+1}|\{z\}_0^k) = \int p(x_{k+1}|x_k)p(x_k|\{z\}_0^k)dx_k \\ &\underline{\text{Update}} \text{ likelihood weighting} \\ &p(x_{k+1}|z_{k+1}, \{z\}_0^k) = p(x_{k+1}|\{z\}_0^k)\frac{p(z_{k+1}|x_{k+1})}{p(z_{k+1}|\{z\}_0^k)} \end{aligned} \tag{4.5}$$

Applying Eq. 4.4 and 4.5 to a second order linear control system with Gaussian random variables is a Linear-Quadratic-Gaussian (LQG) problem, probably the most fundamental optimal control problem [113]. In particular, the famous Kalman filtering is a closed form equation of this problem [113].

In a general distributed sensor network, the measurements are not equally useful and incur different resource expenditures; the system must be nonlinear. Choosing the optimal set of sensors and their measurement process leads to intractable computation. The authors

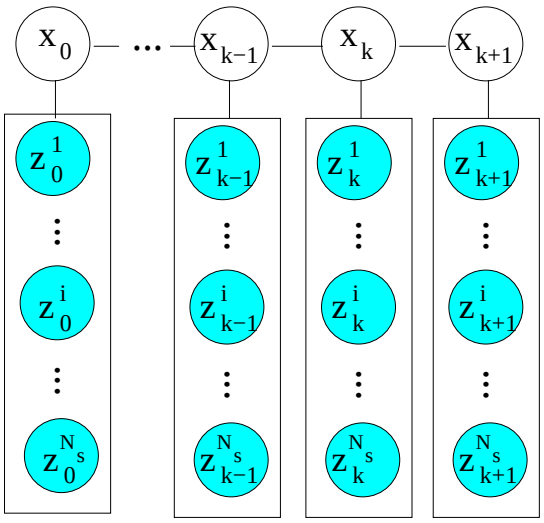


Figure 4.3: A sensor network for dynamic estimation.

in [112] give a detailed analysis of sensor collaboration in ad hoc sensor networks using an information-driven approach. They also present performance bounds and approximation algorithms.

4.1.3 Fundamental Bounds on Information Gathering Systems

Decentralized sensor networks tradeoff the communication requirement over the network with less information available to the fusion center [108]. Luo and Tsitsiklis consider the problem of minimizing the amount of communication in decentralized estimation with two sensors [114]. For systems that only involve Gaussian random variables, they obtain a tight lower bound on the number of messages that the fusion center must receive from the sensors to evaluate the final results. When the system is linear, these bounds are effectively computable. Aldosari and Moura adopt a non-asymptotic approach and optimize both the sensing and fusion sides with respect to the probability of detection error [109]. Their work help us understand tradeoffs between sensor network parameters like number of sensors, degree of quantization at each local sensor, and SNR.

Section 4.1 reviews several topics in decentralized detection. Research results from these areas guided our development of a distributed cooperative spectrum sensing system needed in the prototype DSA network. For example, Tsitsiklis's work helped us choose fusion rules to process sensor data streams sent by multiple sensors. Further, our future work will apply distributed estimation methods to locate a primary signal's geo-location.

4.2 Cooperative Spectrum Sensing for DSA

The goal of DSA is to increase spectrum utilization in three dimensions: frequency, time, and space. To guarantee non-interference with incumbent users, spectrum sensors must quickly detect any radio environment variation with attributes on those three dimensions. Section 4.1 reviews decentralized detection, mainly from a hypothesis test perspective, which can be used to formulate methods for primary user signal detection.

However, cooperative spectrum sensing for DSA also needs to provide a radio environment map; this is more about sensing semantic variables which contain heterogeneous and unstructured information [115]. The fusion center must interpret the context and meaning of sensor data and provide a coherent view of the radio environment. Further, such a sensor network has particular performance requirements on data latency and data reliability. Delayed or erroneous data error may cause secondary users to interfere with incumbent users because secondary users could not be informed quickly or reliably to switch to other spectrum whitespace upon incumbent users' appearance. Therefore, the overall goal of cooperative spectrum sensing is to quickly and reliably provide secondary users with necessary information about spectrum environment, which covers spectrum information in three dimensions: frequency, time, and space. Any cooperative spectrum sensing design should achieve such goals, while attacking challenges faced by most other types of sensor networks like resource usage optimization and security [43].

From the network structure perspective, a cooperative spectrum sensor network may be centralized, for example in IEEE802.22 [29], or distributed, as in DARPA's XG project [77] and Wireless Network after Next (WNaN) project [116]. In the former case, cooperative sensor data is fused at special data centers and sensor data communication goes through control channels [117]. If those data centers also manage spectrum access by secondary users according to some pricing rules, they are called spectrum brokers [118]; particularly when their pricing is considered for dynamic spectrum access.

In the distributed case, some sensor nodes, usually anchored on secondary users, fuse neighboring sensors' data in determining spectrum whitespace; an in-band data channel or a dedicated control channel can be used for sensor data communication [119]. In either case, DSA involves spectrum sensing, sensor data communication, data fusion and storage, and management of secondary users' access of spectrum. Because the DSA paradigm puts stringent conditions on both data latency and data reliability in the above network, special consideration is needed not only at the network level for routing management, but also within all involved end nodes for sensor data process and management.

4.2.1 Semantic Sensor Data

The fusion center in cooperative spectrum sensing must provide a semantic radio environment map for secondary users to opportunistically use vacant spectrum. This part of the dissertation constructs such a map from the perspective of sensor capabilities.

A simple energy detection result is shown in Figure 4.4. As we can see, it presents information in the frequency domain. If the sensor is continuously running, the timing dimension can be derived. Cooperative sensors using location techniques can locate incumbent users given that sensors have their own relative or global locations [120]. Moreover, as discussed in Chapter 3, signal classification can recognize the type of a signal. Combining all the above information, a basic radio environment map directly constructed from signal sensors may look like Table 4.1. Of course, more advanced signal sensors can provide more semantic data about the radio environment like a signal's duty cycle and even symbol rate. In particular, information inherent in this table includes available vacant channels – those frequency ranges within which no signal is detected.

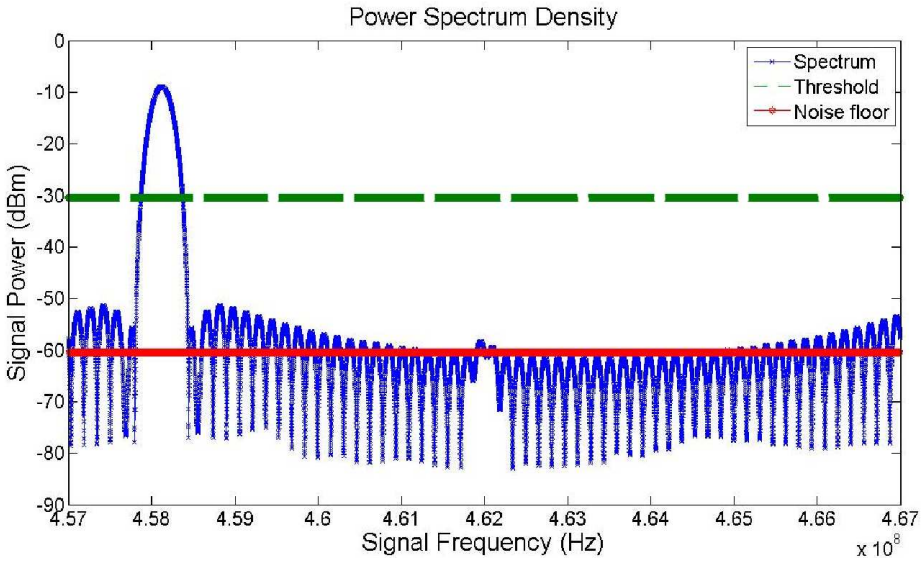


Figure 4.4: A power spectral density. ©2009 IEEE. Reprinted, with permission, from Ge et al., "A cooperative sensing based spectrum broker for dynamic spectrum access," IEEE Military Communications Conference (MILCOM), Boston, MA, 2009.

4.2.2 Spectrum Sensing and Data Processing in Sensors

To achieve the goal of both low data latency and high data reliability in a cooperative spectrum sensing network, each sensor needs a special consideration of its data processing and management. An enabling system is designed and implemented in this section.

Table 4.1: An example of required spectrum information for DSA applications.

ID	F_min (MHz)	F_max (MHz)	Power (dBm)	Modulation	SNR (dB)	Time(y:m:d:h:m:s)	Location
1	462	464	2	BPSK	20	2009:5:6:15:21:12	(λ_1, φ_1)
2	470	471	1	None (Unknown)	17	2009:5:6:15:21:45	(λ_2, φ_2)
3	466	470	4	8PSK	23	2009:5:6:15:20:56	(λ_3, φ_3)
...							

For the discussion purpose, the typical sensor architecture is again shown here in Figure 4.5; its components are introduced in Section 3.1.1. To perform spectrum sensing, the sensing unit acquires radio signals and sends them to the processing unit. The processing unit calculates required signal information: such as spectrum power density, bandwidth, and center frequency. The transceiver then transmits the processed sensor data. The location system is used to determine the sensor's location.

First of all, the sensing unit can adopt a fixed sensing period in spectrum sensing. However, it is hard to decide the right sensing period. If it is too short (a higher duty cycle), the sensing unit will consume too much power; if it is too long, sensors may miss detecting radio environment variations which result in interference to incumbent users from secondary users' transmission. Considering incumbent users' behavior, dynamic sensing scheme can be used so that incumbent users' behavior pattern can be utilized. For example, in the cellular band, there are many short cellphone calls and the remaining are spread over a "semi-heavy" tail [76]. In this case, sensors can adopt an initially rapid sense for few seconds, and then less frequent sense of the same band, plus opportunistically sensing to detect abnormal patterns.

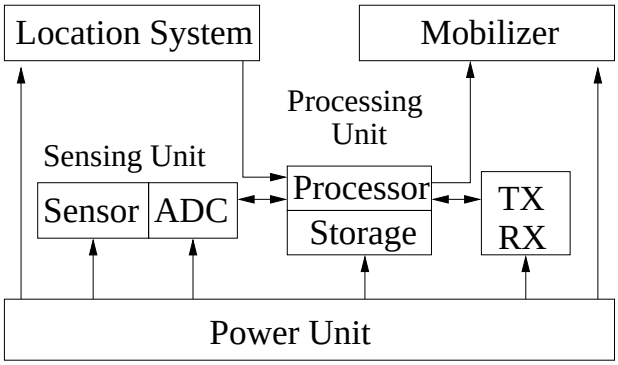


Figure 4.5: A typical sensor architecture.

Second, sensors can be designed as reactive or proactive [121]: proactive sensors pe-

riodically sense the environment and send out sensed data; reactive sensors react immediately to sudden and drastic changes in the value of a sensed attribute. In the DSA situation, neither scheme is suitable. The proactive scheme results in redundant data communication overhead when the radio environment does not change so quickly. The reactive scheme on the other hand is not robust enough to differentiate between sensor failure or an unchanged environment, therefore losing data credibility. Instead, the author selects a hybrid scheme by which nodes not only respond to sudden changes in sensed attribute, which is reactive, but also send data periodically, which is proactive. The strategy is shown in Figure 4.6.

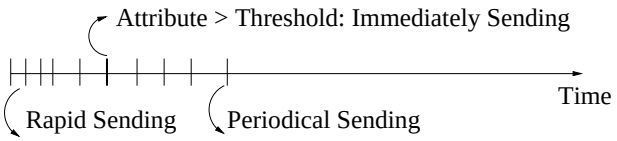


Figure 4.6: Spectrum sensor data update strategies.

Third, the sensor can be *stateless* which means that any sensing result will be sent out immediately without storage, or *stateful* which means that sensing result will be stored in the sensor for some time duration. Considering that both low data latency and high data reliability are required in cooperative spectrum sensor network, redundant data communication must be reduced or avoided. Under conditions that the radio environment doesn't change so often, periodical spectrum sensing results in the same data being sent out. Most of the time this is a waste of network bandwidth and may cause data communication delay when congestion happens. This problem can be alleviated if sensors are *stateful*.

Combining all the above strategies, a data process and management system is shown in Figure 4.7. Given a sensing frequency f_s (the number of times per second that the sensor senses), T is chosen as the maximum period after which the sensor must update its status to a data fusion center. Any real-time data sent from the sensing unit is compared with previously stored data in the local database, if the difference is larger than a threshold (Δ), the new data will be updated to the data fusion center and the counter indicating the sensor's inactive duration is reset. If the difference is less than Δ, the counter is increased by one. If the counter is larger than $M = T \times f_s$, the sensor will update its status to the data fusion center too. But only a short message is sent out to the data fusion center indicating that this sensor is still alive and it detects the same value. Each time new data is detected, it will not only update the data fusion center but also the local database.

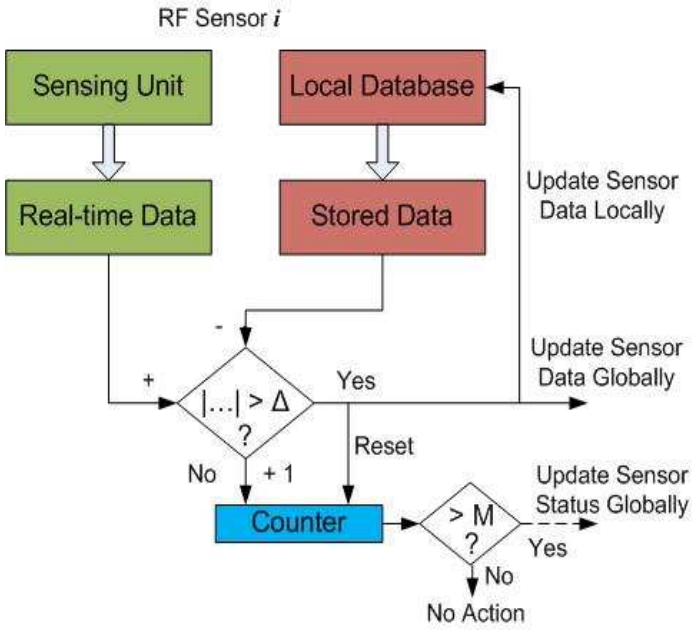


Figure 4.7: The scheme for data flow and processing within a sensor. ©2009 IEEE. Reprinted, with permission, from Ge et al., "A cooperative sensing based spectrum broker for dynamic spectrum access," IEEE Military Communications Conference (MILCOM), Boston, MA, 2009.

4.2.3 Data Processing in Spectrum Servers

The goal of cooperative sensing in DSA is to provide real-time spectrum environment information in three dimensions: frequency, time, and space. Therefore, sensors have to detect related radio environment variations (as discussed in Section 4.2.1) and update processed data to a data fusion center – the spectrum server in this dissertation. Moreover, the spectrum server's role is not only fusing connected sensors' data, but also deriving geo-locations of detected signals and other required information for spectrum sharing.

During data fusion, a spectrum server first waits for a certain period of time to receive results from all the sensors, it then fuses the received data streams. However, synchronization of such data communications is difficult because sensors may perform actions at different times [43]. For example, in one period, a sensor may timeout its operation and does not send any data to the server. Further, a sensor network may get congested, thus delaying a sensor's data report to the spectrum server. In either case, the credibility of a final decision by the server is questionable.

By following the updating strategies introduced in Section 4.2.2, perfect synchronization is not needed and credibility of sensor data can be still guaranteed. However, the challenge is how to process multiple unsynchronized data streams. To solve this problem, a multi-threading method [122] together with database technology [123] is suitable for the

spectrum server data management. As shown in Figure 4.7 and Figure 4.8, the spectrum server allocates to each sensor a corresponding database table for receiving data from the sensor. Based on updating in these tables and status messages sent by all the sensors, a separate database table is used to update all sensors' status. Sensor data stored in all the tables will be fused in a master database; taking this information together with the status table, any variation in the time domain can be derived.

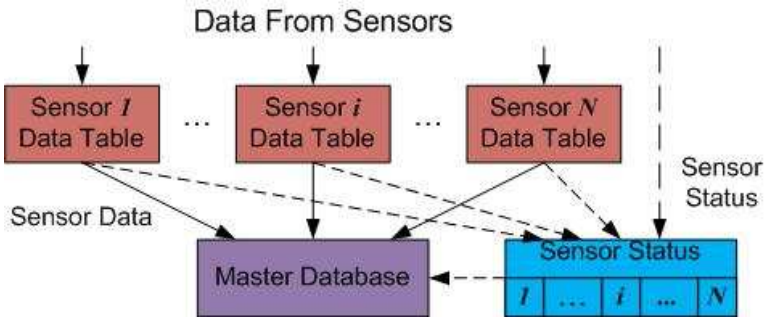


Figure 4.8: Data management in a spectrum server. ©2009 IEEE. Reprinted, with permission, from Ge et al., "A cooperative sensing based spectrum broker for dynamic spectrum access," IEEE Military Communications Conference (MILCOM), Boston, MA, 2009.

Once sensor data is available to the master database, data fusion techniques are needed and they must be robust and fault-tolerant so that they can handle uncertainty and faulty sensor readouts. Sensor data fusion has attracted wide attention [124]. Some existing well-known methods include: probabilistic models and Bayesian data fusion methods as discussed in Section 4.1. Further, the OR logic rule is applied in [105] as the fusion rule and the AND and M-out-of-K rules are considered in [104]. The likelihood ratio test (LRT) based decision is also considered in [104] where it is shown that soft decision fusion has the ability to achieve better performance compared to hard decision combining. The author uses M-out-of-K rules for data fusion in the spectrum server.

In cooperative spectrum sensing, data from identical sensors distributed over a geographical area can be divided into two types. The first type data has same kind of detected information for all sensors, for example, the existence of signal and its frequency range. The other type data is complimentary among sensors, for example, detected signal strength at different locations used to determine detected signal's location by technologies such as triangulation [120]. The final table in the spectrum server may look like what is shown in Table 4.1.

4.3 Secondary User Network

4.3.1 Data Management and Secondary User Access in DSA Broker

A DSA broker bridges the cooperative spectrum sensing network and the secondary user network. It derives necessary spectrum information from spectrum servers for secondary users' spectrum sharing. It also manages secondary users' dynamic spectrum access. Using MySQL database technology, the author designed a DSA broker architecture as shown in Figure 4.9. The processing core has to identify vacant spectrum bands and differentiate active signals between incumbent users and secondary users. The processing core derives necessary spectrum information for secondary users' spectrum sharing. However, secondary users must first register with the DSA broker. The DSA broker also sends alarms to registered secondary users upon incumbent users' presence so that secondary users can switch to other vacant channels.

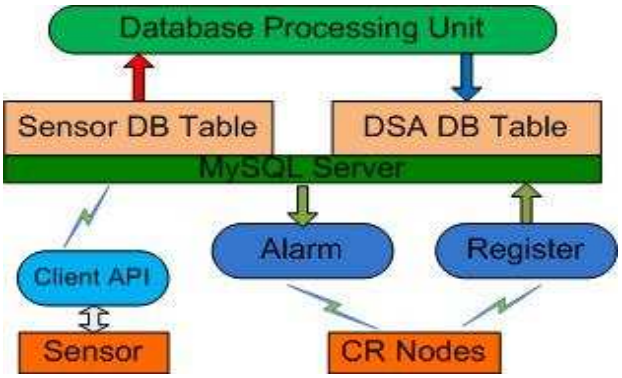


Figure 4.9: The DSA broker architecture. ©2009 IEEE. Reprinted, with permission, from Ge et al., "A cooperative sensing based spectrum broker for dynamic spectrum access," IEEE Military Communications Conference (MILCOM), Boston, MA, 2009.

To differentiate active signals between incumbent users and secondary users, the DSA broker has a database of incumbent user allocation by regulatory bodies such as the FCC. The DSA broker also includes a real-time secondary user database derived from the registration mechanisms. Any detected signal will be compared with the incumbent user signal type to define its user type. An example of such a data table is Table 4.2 To reduce processing burden, the event trigger mechanism is used so that the table is updated only when the spectrum server is newly updated.

Secondary users have to register with the DSA manager to access the necessary information for spectrum sharing. Registration information could include occupied bandwidth, center frequency, power, geo-location, MAC address, IP (if available), and usage time duration. An example is shown in the "DSA broker cognitive radio registry table" of Fig-

Table 4.2: An example database in the DSA broker.

ID	F_min (MHz)	F_max (MHz)	Pow (dBm)	Mod	SNR (dB)	Time(y:m:d:h:m:s)	Loc	Busy	Signal Type
1	462	464	2	BPSK	20	2009:5:6:15:21:12	(λ_1, φ_1)	0	None
2	470	471	1	None	17	2009:5:6:15:21:45	(λ_2, φ_2)	1	2nd
3	466	470	4	8PSK	23	2009:5:6:15:20:56	(λ_3, φ_3)	1	Primary
...									

ure 4.13. Upon incumbent users' presence, secondary users will be informed by the DSA manager to switch to available vacant channels.

4.3.2 Spectrum Usage Statistics

The feasibility and success of spectrum sharing depends crucially on if, when, and how idle spectrum becomes available. For secondary users that use spectrum sensing technologies in DSA, how easily idle spectrum can be sensed is also a critical factor. For example, TV broadcasting is either idle or active, both over long periods of time. In contrast, cellular spectrum usage exhibits much more variations both in time and space [76]. Therefore, more agile DSA techniques are needed for sharing cellular band spectrum than the TV broadcasting spectrum. Further, communication data rates between secondary users opportunistically using the above two types of spectrum band may vary quite differently. The data rate in the TV band may be much more stable than the rate in the cellular band, therefore, the "quality" of idle spectrum within the two bands is different too. Idle spectrum over other bands may demonstrate similar or even more diverse usage patterns. Some statistics values and even models are helpful to differentiate idle spectrum bands with different usage behaviors.

We designed and developed *stateful* DSA brokers so that they can store durations of spectrum usage data. For example, a frequency range can be divided into different bands and each band has a table which continuously updates not only each channel's availability, but also its usage statistics. The stored data can be compiled from both measurements and registrations [76]. Based on each channel's table of spectrum usage, secondary users can choose the idle spectrum band which best matches their application requirements. For example, real-time voice communication needs continuous and stable spectrum availability while normal email checking doesn't need such "good quality" spectrum.

This section provides some examples illustrating such statistics. To calculate spectrum usage statistics based on some spectrum measurement, we can apply a threshold to differentiate between idle and active spectrum. The threshold selection may depend on cooperative spectrum sensors' performance and other DSA technologies. We can differentiate between incumbent usage and secondary usage using sensed data and secondary spectrum usage registration. A spectrum usage measurement is shown in Figure 4.10 which was taken in a hotel in downtown Chicago in March 2008. A spectrum usage map post-processed by thresholding may look like Figure 4.11, where active spectrum blocks are either used by incumbent users or secondary users.

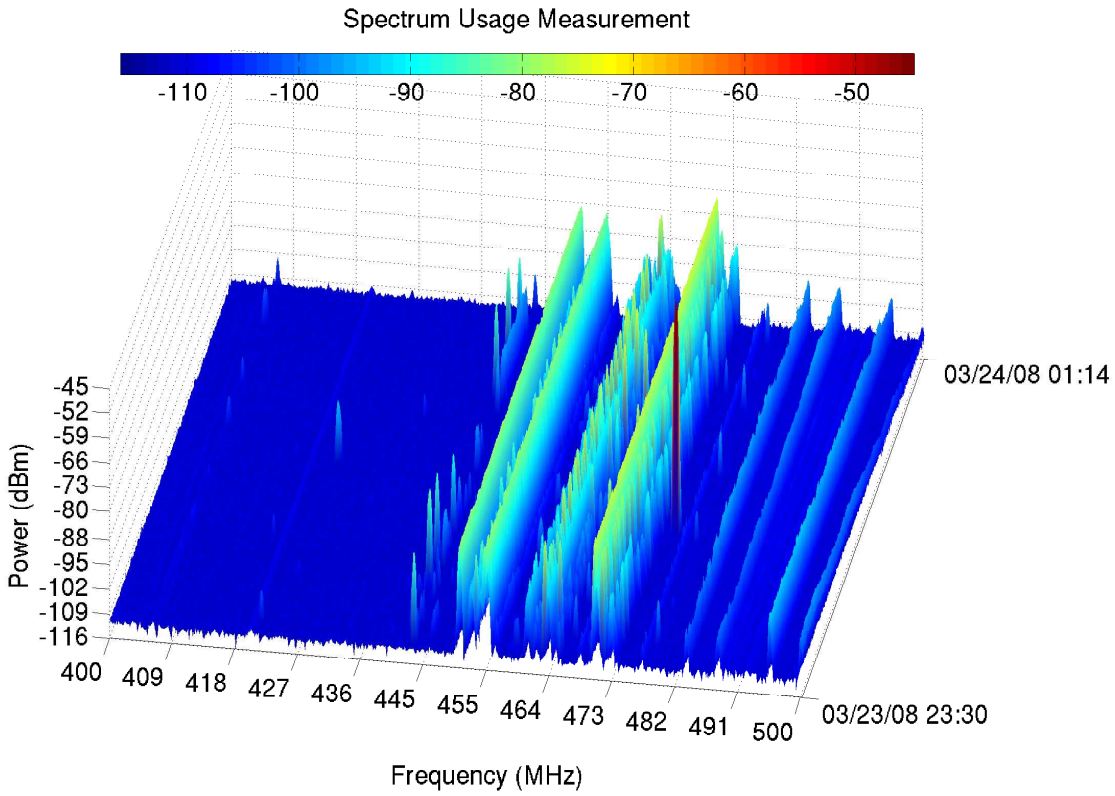


Figure 4.10: Spectrum usage measurement, courtesy of Thomas Rondeau and Keith Nolan.

At a specific geo-location and within time duration from t_0 to t_1 and frequency band from f_0 to f_1, spectrum usage efficiency solely by incumbent users is defined and computed by Eq. (4.6).

$$\text{Eff}_{\text{incumbent}} = \frac{\sum_1^m (I_i)}{(t_1 - t_0) \times (f_1 - f_0)} \tag{4.6}$$

where I_i is the smallest spectrum block (with the unit of time $\times$ frequency) used by some

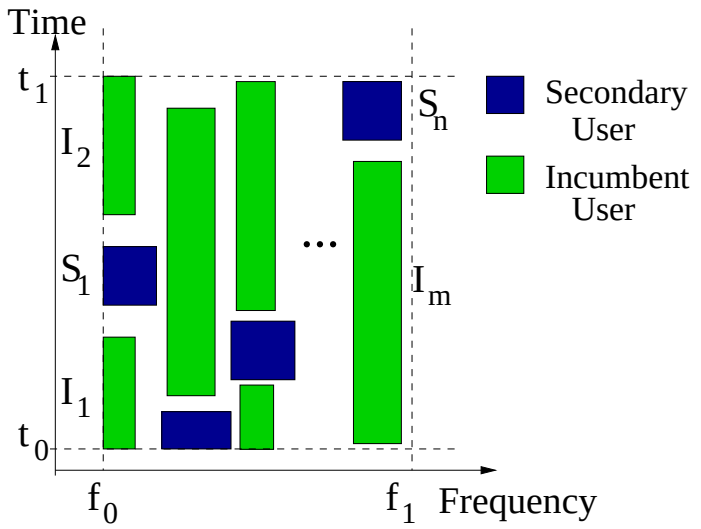


Figure 4.11: An example of simplified spectrum measurement. ©2009 IEEE. Reprinted, with permission, from Ge et al., "A cooperative sensing based spectrum broker for dynamic spectrum access," IEEE Military Communications Conference (MILCOM), Boston, MA, 2009.

incumbent user within a time and frequency range. Similarly, S_i is the spectrum block used by secondary users. m is the total number of active spectrum blocks used by the incumbent users.

The overall spectrum usage efficiency including both incumbent users and secondary users will be decided by Eq. (4.7).

$$\text{Eff}_{\text{overall}} = \frac{\sum_{1,1}^{m,n}(I_i + S_j)}{(t_1 - t_0) \times (f_1 - f_0)} \tag{4.7}$$

where S_i is the smallest spectrum block unit used by some secondary user within a time and frequency range. n is the total number of active spectrum blocks used by the secondary users.

Considering that the memory size of the DSA broker is finite, it can only store spectrum usage information for a finite time interval as shown in Figure 4.11. Therefore, we can use the following formula to update the efficiency from t_0 to t_2 given that $\text{Eff}_{\text{incumbent}}^{t_0 \sim t_1}$ is known, for example for incumbent usage efficiency without storing all the spectrum usage information from t_0 to t_2.

$$\text{Eff}_{\text{incumbent}}^{t_0 \sim t_2} = \frac{(t_1 - t_0)(f_1 - f_0)\text{Eff}_{\text{incumbent}}^{t_0 \sim t_1} + \sum_1^m (I_i)^{t_1 \sim t_2}}{(t_2 - t_0) \times (f_1 - f_0)} \tag{4.8}$$

Some dynamic spectrum utilization statistics are shown in Figure 4.13.

Further, statistical models can be used to represent spectrum usage patterns within frequency and time domain. For example, in a cellular network, incumbent users' call duration is modelled as an exponential distribution, though deviations were reported in [76]. The above spectrum usage statistics can be very useful in spectrum trading too because it characterizes the "quality" of vacant spectrum band [118].

The above spectrum usage statistics can be very useful in spectrum trading. too. The registration mechanism introduced in Section 4.3.1 allows spectrum licensees to sublease spectrum to secondary users. Spectrum trading promises to increase the incentive of spectrum sharing [11, 125] because it can stimulate users to sell and lease under-utilized spectrum. For example, auctions usually promise efficient allocation of scarce resources [126]. However, necessary information is required to carry out a spectrum auction. Most current research on spectrum trading is primarily based on information in the dimension of frequency [126, 118]. Such models do not fully capture the essence of DSA: increase spectrum efficiency in three dimensions (frequency, time, and space). Pricing should be based on the integral value of spectrum within the above three dimensions. For fairness, the spectrum information needed for auction should contain statistics of spectrum usage.

4.4 Experiment

To demonstrate the cooperative spectrum sensing network, the author (and his colleagues) designed a test network as shown in Figure 4.12. Four different sensors were built. The first sensor employs broadband parallel RF sensing on a Lyrtech "Small Form Factor" (SFF) Software Defined Radio (SDR) board; this sensor's algorithm primarily runs on the embedded FPGA and can cover signals up to 20MHz in bandwidth. The second sensor, with a Universal Software Radio Peripheral (USRP) as the front end, implements a signal classification and synchronization system using frequency and time domain signal features; it is able to classify analog AM and FM, digital M-ary PSK, M-ary QAM, M-ary FSK, and OFDM signals [127]. The third sensor is a broadband energy detector based on an Anritsu MS2781A Signature Signal Analyzer; this sensor can cover signals up to 20MHz bandwidth. The fourth sensor uses narrow band energy detection followed by a k-nearest neighbor (K-NN) algorithm classifying signal features on a laptop with a USRP as the RF front end; it is able to classify analog FM and AM, digital BPSK, QPSK, and 4FSK signals [83]. Wi-Fi adapter cards were used to join these sensors into a sensor network. Please note that no CDMA signal was used in our experiment because our signal sensors can't classify an unknown CDMA signal without knowing the spreading code. Further, two reconfigurable SDR radios based on GNU Radio and USRP were used to simulate secondary users,

which can reconfigure and switch transmitting frequency agilely. One secondary node was mounted with sensor 4. These two secondary user nodes communicated with each other using vacant channels obtained from the DSA broker.

In a lab environment, the author used Family Radio Service (FRS) walkie talkie radio and USRP based SDR in simulating a dynamic spectrum environment over the frequency range from 460MHz to 468MHz. The four sensors continuously detected spectrum environment and updated the spectrum server with detected results. Four sensors covered a user specified frequency range while continuously updating the spectrum server with detected signal features. The spectrum server fused sensor data and stored them in its "main sensor database" as shown in Figure 4.13, a user friendly interface was developed to monitor the DSA broker. Semantic sensor data includes starting frequency ("F_{min}"), ending frequency ("F_{max}"), power, modulation, the time being detected, and SNR of each signal. Unidentified signal modulation is specified with NULL. Based on sensors' result and its knowledge database of incumbent users, the DSA broker decides whether each channel is an incumbent signal or secondary user. Further, the DSA broker also segments spectrum into occupied or unoccupied frequency bands by showing as 1 or 0. The total table is shown in the "DSA broker bandwidth table" in Figure 4.13. Specifically, the color bars indicate real-time spectrum availability within a user specified frequency range: green color means active signal bands and red color means vacant frequency bands. Spectrum usage statistics are also displayed.

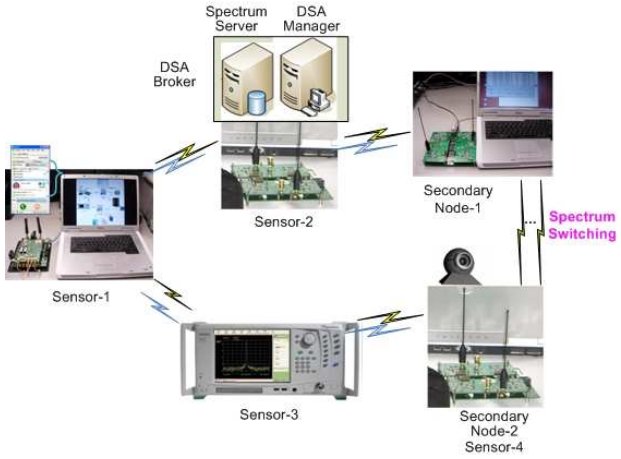


Figure 4.12: Our overall experiment setting. ©2009 IEEE. Reprinted, with permission, from Ge et al., "A cooperative sensing based spectrum broker for dynamic spectrum access," IEEE Military Communications Conference (MILCOM), Boston, MA, 2009.

Secondary users interfaced with the DSA broker through a Wi-Fi based control channel and registered with the DSA manager with their IP address, center frequency, modulation,

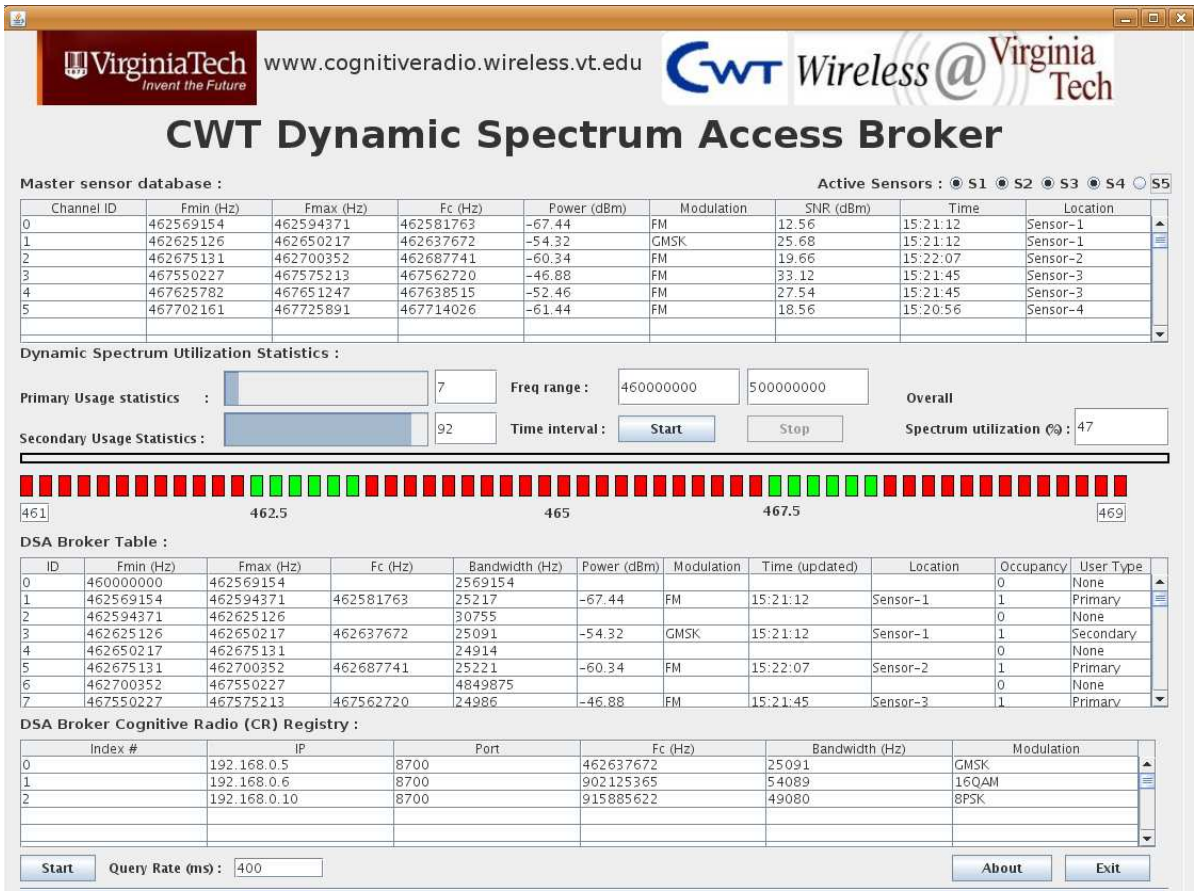


Figure 4.13: The DSA broker graphic user interface. ©2009 IEEE. Reprinted, with permission, from Ge et al., "A cooperative sensing based spectrum broker for dynamic spectrum access," IEEE Military Communications Conference (MILCOM), Boston, MA, 2009.

etc., as shown in the "DSA broker cognitive radio registry" in Figure 4.13. Secondary users chose vacant channels through the "DSA broker table" to communicate with each other. Note that the current DSA broker is not equipped with location technologies, and it simply chooses the name of a sensor that detects the signal. The DSA Broker broadcasted alarms to secondary users through the Wi-Fi control channel when incumbent signals appeared, then secondary users chose new vacant channels to continue previous communication. The whole network was demonstrated in the 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN 2008) in Chicago.

4.5 Conclusion

This chapter first overviews progress made in decentralized detection. It then gives a set of schemes for spectrum data flow and process management within spectrum sensors and

DSA brokers, which are used in a cooperative spectrum sensor network requiring low data latency and high data reliability. Further, this chapter presents some spectrum usage statistics, which are not only useful for secondary users' spectrum sharing, but also necessary in any spectrum trading. Finally, a network experiment of four spectrum sensors with two SDR based secondary users is given.

Chapter 5

Decentralized DSA Networks: A Prototype Design and Experimental Results

Significant research progress has been made in cognitive radio and DSA networks since Mitola's seminal dissertation [17]. For example, many measurements of current spectrum utilization are available. Theoretical analyses and computational simulations of DSA networks also abound. In sharp contrast, few network systems, particularly those with a decentralized structure, have been built even at a small scale to investigate the performance, behavior, and dynamics of DSA networks under different scenarios.

On the other hand, supporting DSA in decentralized wireless networks drives the complexity both within individual nodes and of the overall network to an unparalleled level. Therefore, experiments are necessary to gain insights into network design and to investigate network performance under different scenarios [31]. Further, in research of complex systems like cognitive DSA networks, experimental methods can access system/network conditions that may be neglected by either theoretical analysis or computational simulation.

The author designs a decentralized and asynchronous DSA network and builds a ten-node network prototype based on software radio technologies, signal detection and classification methods, distributed cooperative spectrum sensing systems, modified mobile ad-hoc network (MANET) protocols, a multi-channel allocation algorithm.[1] This chapter details the network's design and implementation as well as its essential functions. Through systematic experiments, the author identifies several performance determining factors for

[1]The multi-channel network development was collaborative work between the author and his colleague Mustafa Y. El-Nainay.

developing decentralized DSA networks. For the purpose of easy access and being self contained, some figures shown in previous chapters are reproduced here.

5.1 Introduction

The author's view of decentralized DSA networks is shown in Figure 5.1. In this network, nodes opportunistically use vacant channels when incumbent users are not transmitting. The whole network consists of decentralized mobile nodes and works in a dynamic radio environment where fading and interference exist and incumbent users may appear at any time. To increase network battery life and reduce network communication overhead, the network is properly partitioned into clusters [43], each cluster dynamically activates its associated nodes' functions like signal detection and classification, data fusion, and data relay. Distributed cooperative spectrum sensing happens with each cluster and across neighboring clusters. This chapter presents the network prototype corresponding to Figure 5.1. In particular, the current prototype aims at asynchronous and decentralized DSA networks. This network enables DSA by using reconfigurable radio nodes, signal detection and classification methods [127], distributed cooperative spectrum sensing schemes [128], distributed dynamic wireless protocols, and intelligent algorithms at both node and network levels [129]. The current prototype uses a control channel to support distributed cooperative sensing and network management. The following sections introduce the enabling functions and systems of this network prototype and present experimental results and lessons.

5.2 Cooperative Spectrum Sensing

To guarantee non-interference to incumbent users, spectrum sensors in a DSA network must detect signals event at low signal-to-noise ratio (SNR) and this performance should still hold up under dynamic and hostile conditions like shadowing, fading, interference, and multi-path [42, 44]. However, a single signal detector may fail to achieve a good performance under such conditions. Cooperative spectrum sensing [42] can solve the above problems by geographically distributing several sensors and it is widely endorsed for DSA networks [42, 104, 105]. Furthermore, cooperative sensing can improve detection reliability and reduce sensitivity requirements for single sensors by exploiting their cooperative gains [42].

Chapter 4 details the cooperative spectrum sensing system; this section summarizes some key points and focuses on integrating it into the DSA network prototype.

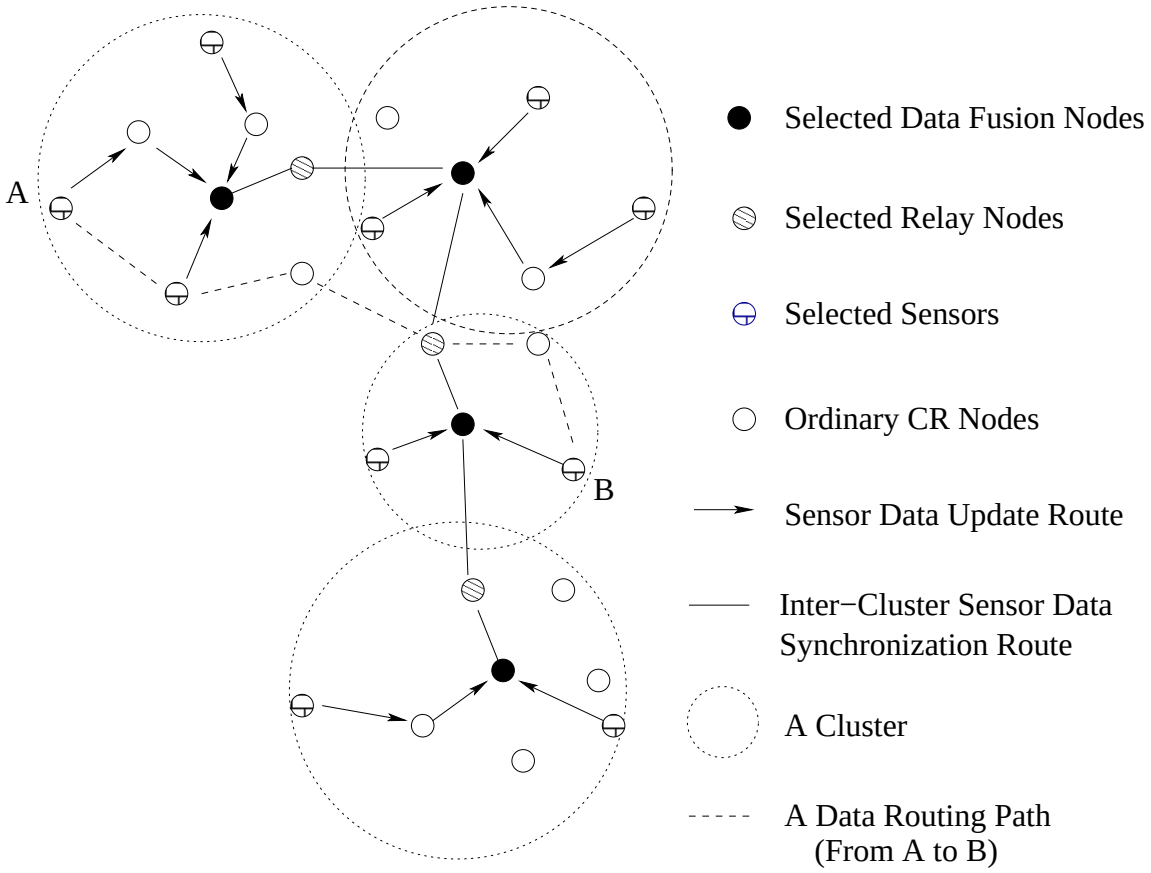


Figure 5.1: The author's view of a decentralized DSA network.

5.2.1 Signal Sensors

The sensors used in the network prototype are realized by GNU Radio [46] and USRP version I [50]. The USRP is attached to a laptop, which uses a Wi-Fi chip as the control channel transceiver and serves as the processing unit and power unit. Wide band energy detectors and narrow band signal classifiers are used in the network prototype. The receiving gain of each sensor is tuned within the effective dynamic range of USRP's RF front end.

Energy Detection

Whenever a DSA network has to switch to new channels, it must establish that incumbent users are not present in those channels. Energy detection has the advantage of identifying suitable channels over a wide frequency range. Moreover, it can use efficient algorithms on a simple system with minimum power consumption.

Fast Fourier Transform (FFT)-based energy detection is used as a signal detection method. Each detector uses a USRP to collect signal samples in a passband and calcu-

lates the discrete power spectral density (PSD) with appropriate windowing and noise irregularity reduction. It then compares the PSD data with a pre-defined minimum threshold for energy detection. Further, the detector also calculates each active signal's bandwidth, center frequency, power, SNR, and the time that it first became active.

Signal Classification

A simple energy detector cannot differentiate between signals transmitted by incumbent users from those by peer secondary users. A quiet period is proposed in [119] to enable spectrum sensing, but it requires strict synchronization coordination among DSA nodes to guarantee that no single node is transmitting when spectrum sensing happens. Signal classification is necessary in unsynchronized decentralized DSA networks because it can identify incumbent users even when signals from secondary users exist.

The current DSA network prototype applies the Universal Classifier and Synchronizer (UCS) algorithm [85] in signal classification. UCS uses a radio signal's time and frequency domain features to classify and synchronize a receiver to it and to provide all parameters needed for physical layer demodulation, without knowing any prior modulation information. The current UCS is able to classify AM, FM, MPSK, QAM, MFSK and OFDM modulations based on over-the-air radio signal samples collected either by the GNU Radio/USRP or the Anristu Signature Signal Analyzer [85]. Its theoretical analysis and detailed performance results are available in [85].

The current network prototype requires a signal classifier only to recognize an active signal's modulation. However, signal classification usually involves much higher computational complexity and resource consumption than simple energy detection [44]. As a result, recognizing a signal usually takes a longer time than detecting this signal. A cooperative sensor network usually activates only a necessary small number of signal classifiers to reduce power consumption and increase the network life.

Local Sensor Data Flow Management

To reduce unnecessary data transmissions cooperative spectrum sensing, as discussed in Chapter 4, sensors are designed to be *stateful* – they remember their previous sensing results and update the data fusion node only when they detect a radio environment variation. The data update scheme is shown in Figure 5.2. Given a sensing frequency f_s (the number of times per second that the sensor senses), T is chosen as the maximum period after which the sensor must update its status to a data fusion node. Any real-time data from the sensing unit is compared with previously stored data in the local database. If the difference is larger

than a threshold (Δ), the new data will be updated to the data fusion center and the counter indicating the sensor's inactive duration is reset. If the difference is less than Δ, the counter is increased by one. Currently Δ is a tuple indicating the difference of a signal's frequency range and its modulation.

If the counter is larger than $M = T \times f_s$, the sensor will update its status to the data fusion center too. But only a short message is sent to the data fusion center indicating that this sensor is still alive and still detects the same value. Each time new data is detected, it will update both the data fusion center and the local database.

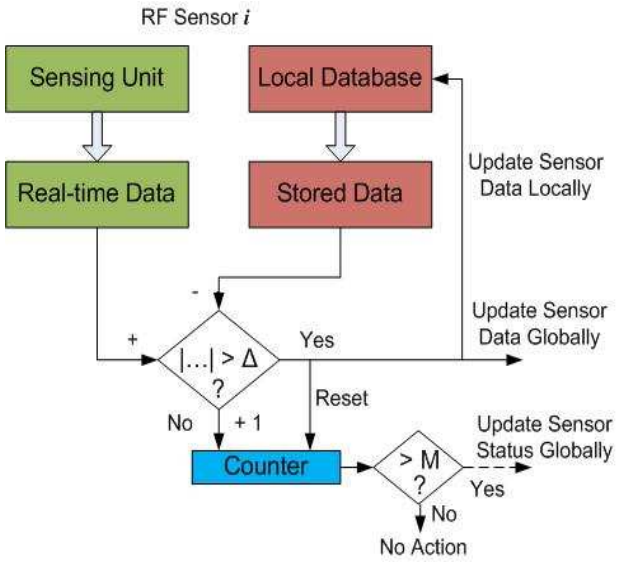


Figure 5.2: The scheme for data flow and processing within a sensor. ©2009 IEEE. Reprinted, with permission, from Ge et al., "A cooperative sensing based spectrum broker for dynamic spectrum access," IEEE Military Communications Conference (MILCOM), Boston, MA, 2009.

5.2.2 Cooperative Sensing System

As stated in Chapter 4, a cooperative spectrum sensing network has particular performance requirements for sensor data latency and reliability in DSA applications. Delayed or faulty data may cause secondary users to interfere with incumbent users because secondary users could not be informed quickly or reliably to switch to other spectrum whitespace upon incumbent users' appearance. On the other hand, false detection or classification of vacant channels may result in unnecessary channel switching. A set of schemes is designed for spectrum data flow and process management within RF signal sensors and data fusion nodes in reducing data latency and achieving high data reliability [128].

DSA Broker

In cooperative spectrum sensing, data from cooperating sensor nodes can be fused in a single node in decentralized networks. The group of all these nodes is usually called a cluster and the data fusion node is usually called a cluster head [121]. The cluster head then broadcasts results to other nodes. There are schemes to determine the cluster head; for example, each node can take turns so that energy consumption is equally distributed among all the nodes [121]. Those data fusion nodes are called DSA brokers here because they serve not only as a fusion center, but also as a local network manager for dynamic spectrum access. Their functions include data fusion, radio environment and network topology awareness, channel allocation, dynamic routing calculation, and secondary user management.

By following the sensor data updating strategies introduced in Section 5.2.1, perfect synchronization is not needed and credibility of sensor data over the network can still be guaranteed. However, one challenge for data fusion is how to process multiple unsynchronized data streams. To solve this problem, a multi-threading method [122] is used together with database technology [123] in the DSA broker data management. As shown in Figure 5.2 and Figure 5.3, the DSA broker allocates to each sensor a corresponding database table for receiving data from the sensor. Based on updates in these tables and on status messages sent by all the sensors, a separate database table is used to update all sensors' status. Sensor data stored in all the tables will be fused in a master database; together with the status table, variations of the radio environment can be identified. The master database table contains the radio environment information sensed by cooperative sensors; one example result is shown in Table 4.1.

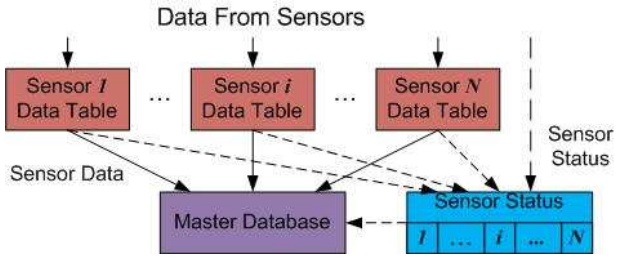


Figure 5.3: Data management in a spectrum server. ©2009 IEEE. Reprinted, with permission, from Ge et al., "A cooperative sensing based spectrum broker for dynamic spectrum access," IEEE Military Communications Conference (MILCOM), Boston, MA, 2009.

Each DSA broker uses a multi-threading based server agent to receive data from sensors, peer DSA brokers, and CR nodes. This ensures that direct database access is operated locally within each DSA broker. Because direct remote database access needs to exchange large data headers and usually takes multiple TCP/IP packet exchanges, these agents can

not only boost network speed, but also significantly reduce network communication overhead. Further, those agents apply data fusion methods to sensor results before updating DSA broker databases. Our current system uses M-out-of-K rules in data fusion [104].

5.3 Distributed DSA Broker Synchronization and Interface with CR Nodes

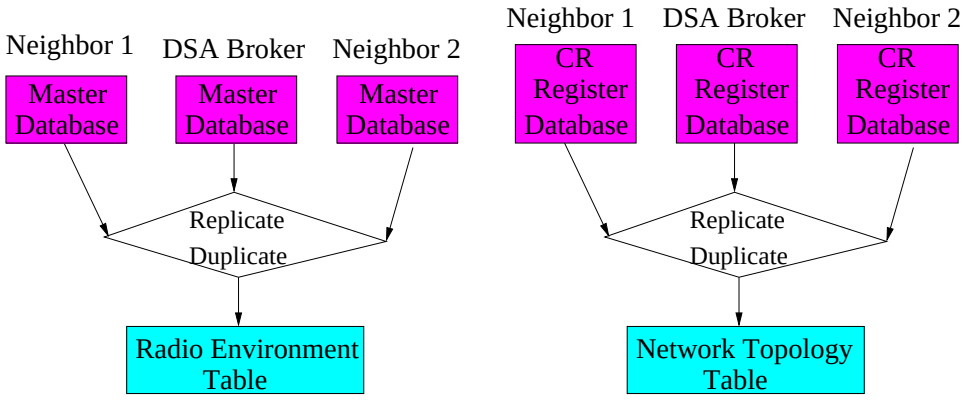
Section 5.2 mainly discusses cooperative spectrum sensing within one cluster. The acquired local radio environment information is enough for cognitive radio nodes within this cluster. However, data communication happens across the whole decentralized network and nodes must agree on common channels to talk to each other. Therefore, inter-cluster DSA brokers need to synchronize their data content in a timely way. Further, if network nodes have MIMO capabilities, multi-channel allocation over the whole network may achieve better network performance. Since multi-channel allocation algorithms are based on network topology, DSA brokers need not only to provide radio environment information, but also to manage CR nodes and collect information about the network topology [54].

5.3.1 Inter-Cluster DSA Broker Synchronization

In the network prototype, each DSA broker registers its associated cluster CR nodes in a registry table. Using registration information with additional necessary technologies like geo-location, each DSA broker can derive the cluster level network topology. To know its neighbor cluster's radio environment and network topology, each DSA broker is synchronized with its neighboring ones, maybe through a few relay nodes for their communications as shown in Figure 5.1. Those possible relay nodes are determined by network level protocols. Essentially these DSA broker nodes are decentralized and distributed over the whole network.

Neighboring DSA brokers synchronize with each other using distributed databases technologies, as shown in Figure 5.4. Two major operations are replication and duplication [130]. Replications are used to look for changes in neighboring databases and make neighboring database look the same. Duplications are used to identify one database as a master and then duplicate that database to its neighbors. The final data is stored in two tables: a radio environment table and a network topology table.

As shown in Table 5.1, the radio environment table shown in Figure 5.4 contains information of the radio environment and vacant channels calculated by a broker algorithm



(a) Synchronize radio environment information

(b) Synchronize network topology information

Figure 5.4: Distributed DSA broker synchronization

agent. Further, this table indicates the user type of occupied channels by comparing detected signals with incumbent ones in existing databases. Finally, it has radio environment information for neighboring clusters. The network topology table mainly consists of information about associated CR nodes within a cluster and its neighbors, for example, geo-location information if available.

Table 5.1: The content of the radio environment table.

ID	F_min (MHz)	F_max (MHz)	Pow (dBm)	Mod	SNR (dB)	Time(y:m:d: h:m:s)	Loc	Busy	Signal Type	Cluster ID
1	462	464	2	BPSK	20	2009:5:6: 15:21:12	(λ_1, φ_1)	0	None (Unknown)	self
2	470	471	1	None	17	2009:5:6: 15:21:45	(λ_2, φ_2)	1	2nd	1
3	466	470	4	8PSK	23	2009:5:6: 15:20:56	(λ_3, φ_3)	1	Primary	3
...										

5.3.2 Island Genetic Algorithm for Multi-channel Allocation

Vu et al. proposed that a DSA based MIMO system (with spatial and frequency multiplexing) can avoid the Gupta-Kumar limit in MANETs [131] and scale linearly with the number of nodes [132]. However, achieving the optimal multi-channel allocation over the whole network is an NP hard problem [33]. Therefore, distributed algorithms are investi-

gated here for multi-channel allocation. A simple scheme is that each node randomly picks a channel among available ones when it transmits data. However, such a scheme usually results in suboptimal performance. For multi-channel allocation, the network prototype applies a distributed learning and optimizing algorithm – the Island Genetic Algorithm (IGA) developed in [54, 33].

Figure 5.5 illustrates the basic idea of the IGA. The IGA divides the population (candidate solutions) into subpopulations that evolve separately each at a different node. Nodes communicate regularly to share their candidate solutions to increase local diversity using a migration policy that defines the migration rate and topology. Network nodes use information shared in the radio environment table and network topology table to apply genetic algorithm operations and migration policies to best allocate available channels to communication links as needed. The Localized IGA used in the current network prototype can also scale to large networks [133].

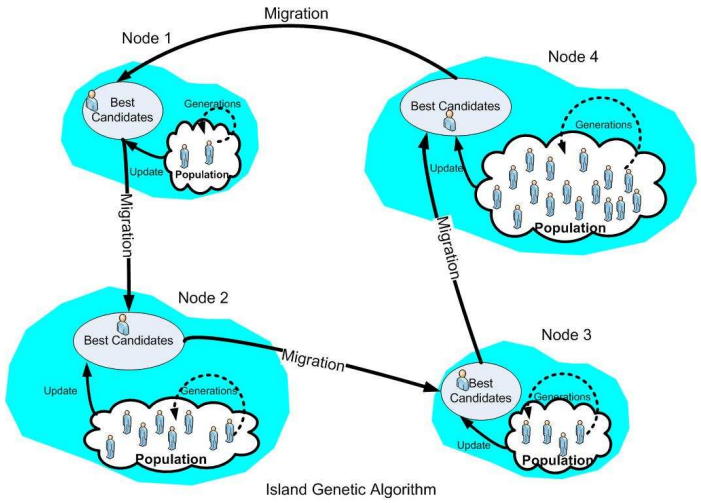


Figure 5.5: Apply Island Genetic Algorithm, courtesy of Mustafa Y. El-Nainay. Reprinted, with permission, from Mustafa Y. El-Nainay, "Island genetic algorithm-based cognitive networks," Ph.D. dissertation, Virginia Polytechnic Institute and State University, 2009.

5.3.3 DSA Brokers Interface with Associated CR Nodes

Section 5.2 and Section 5.3.1 introduce the distributed sensor and DSA broker network which sense variations in the radio environment and compile a coherent table of it. Combining such information with the network topology, DSA brokers apply distributed algorithms to allocate channels for each node in data communication.

Because the radio environment might vary at any time, DSA brokers must quickly notify CR nodes about such variations; meanwhile, network nodes may be mobile and a few

of them are dynamically selected to assume functions like DSA brokers and sensors, those selected DSA brokers must always have the new network topology. The current network prototype provides an interface between DSA brokers and their associated CR nodes for two types of data communications: DSA brokers register their associated CR nodes with information like MAC address, IP address and geo-location if available, and channels and modulation being used; DSA brokers allocate new channels for CR nodes and notify them about new routing information.

The network prototype uses Wi-Fi as a control channel in forming an ad hoc network for data communications described in Section 5.2 and Section 5.3. The network applies Optimized Link State Routing Protocol (OLSR) at the network layer.

5.4 CR Node Architecture and Communication Protocols

Besides functions used in distributed cooperative spectrum sensing, each CR node must work as a secondary user to communicate with the other nodes using sensed vacant channels. Correspondingly, a set of protocols are needed to support such communications.

5.4.1 Overall Node Architecture

As shown in Figure 5.6, the current CR node consists of six components: a learning module, a DSA module, an application controller, a network controller, a SDR system, and a control channel module. The learning module hosts learning functions and algorithms used at both node and network levels. Currently it has a localized IGA for multi-channel allocation at the network level; the author plans to integrate node level learning capabilities used in [20]. The DSA module manages functions used in distributed cooperative spectrum sensing and interfaces with other modules in enabling dynamic spectrum access. Both modules have database components working as knowledge bases. The application controller manages the execution of different applications under different radio and network dynamics. The network controller enables network level protocol configuration and execution; for example, OLSR is listed as one example here. The SDR system supports a reconfigurable MAC and PHY layer. The network prototype relies on a control channel – currently a Wi-Fi system.

The previous sections introduce the learning module and distributed cooperative spectrum sensing. The next material details other node functions and presents how these functions together enable a decentralized DSA network.

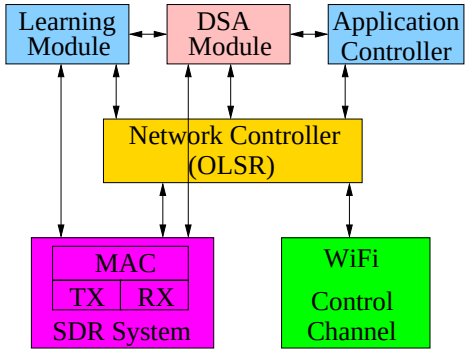


Figure 5.6: The prototype node architecture.

5.4.2 SDR System

The current SDR system relies on GNU Radio and USRP. In particular, the SDR system can receive 3 channels simultaneously and transit on one channel. It uses a frequency multiplex method at the receive side in emulating this 3 by 1 MIMO capability. Further, a MAC layer protocol is designed to control the PHY layer and enable DSA mechanism.

PHY Layer

As shown in Figure 5.7, the PHY layer receives 3 channels simultaneously and reconfigures its transmitter for different transmitting frequencies. More specifically, the USRP down-converts a wide-band RF signal into an IF signal and sends it to GNU Radio in the software domain. The receiver path uses filters to separate this IF signal into three channels. Signal samples in each channel are demodulated and saved in a queue which interfaces with a callback function used by the MAC layer.

However, because USRP's RF front end has a poor non-linearity performance and lacks in filtering separation [134], the transmitter side has a poor performance in frequency multiplexing. Therefore, currently it only supports one transmitting channel. But it is able to reconfigure on-the-fly for switching transmitting frequencies among three channels.

MAC Layer

A carrier sense multiple access (CSMA) based MAC protocol is designed to enable multi-channel communications among CR nodes under DSA environments. This protocol is shown in Figure 5.8. It is exclusively for data communication on vacant channels, not for the control channel.

To understand its working mechanism, let us follow the data chain from the receiving side to the transmitting side. On receiving a data frame, the MAC layer will strip the frame

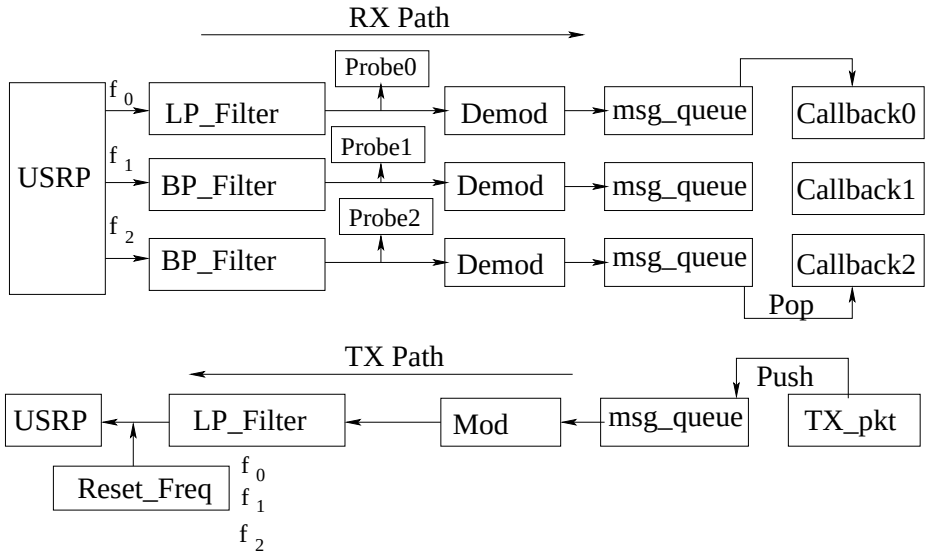


Figure 5.7: Physical layer structure.

header and check its destination MAC address. It only sends to the operating system (OS) data packets for this node and drops others. A received data packet might be for some application at this node, or for relaying data over the DSA network. For the later case, the MAC layer will check its destination IP address and find an appropriate transmitting channel based on a channel allocation table and a routing table. The fragmented data frames are then added with the MAC address of the node for the next hop along a routing path. Finally the data frames are sent out when the node senses that no other nodes are transmitting over the selected channel. If the transmitting side senses that the selected channel is occupied, it waits for a random time interval (determined using the truncated binary exponential backoff algorithm) before trying to send that frame again. After each data frame is transmitted, the MAC enables a much shorter random time backoff so that other nodes do not have to wait for a long time to access the channel.

Under a DSA environment, however, incumbent users may appear at any time. To reduce energy consumption, not all the nodes are allowed to activate the signal classification function. Therefore, a CR node may not be able to realize that a busy channel is actually occupied by incumbent users, instead of by its peer nodes. Solving this problem requires information from the distributed cooperative spectrum sensor network. In doing so, the MAC layer also enables an accumulative time counter of all the backoff time between the interval bounded by transmitting two consecutive data frames. If the whole backoff time exceeds a time threshold t_s, the MAC will signal the application layer to stop sending data and the SDR system to pause its PHY layer. Then this CR node will query the closest DSA broker to confirm whether an incumbent user is present in the current channel. If an

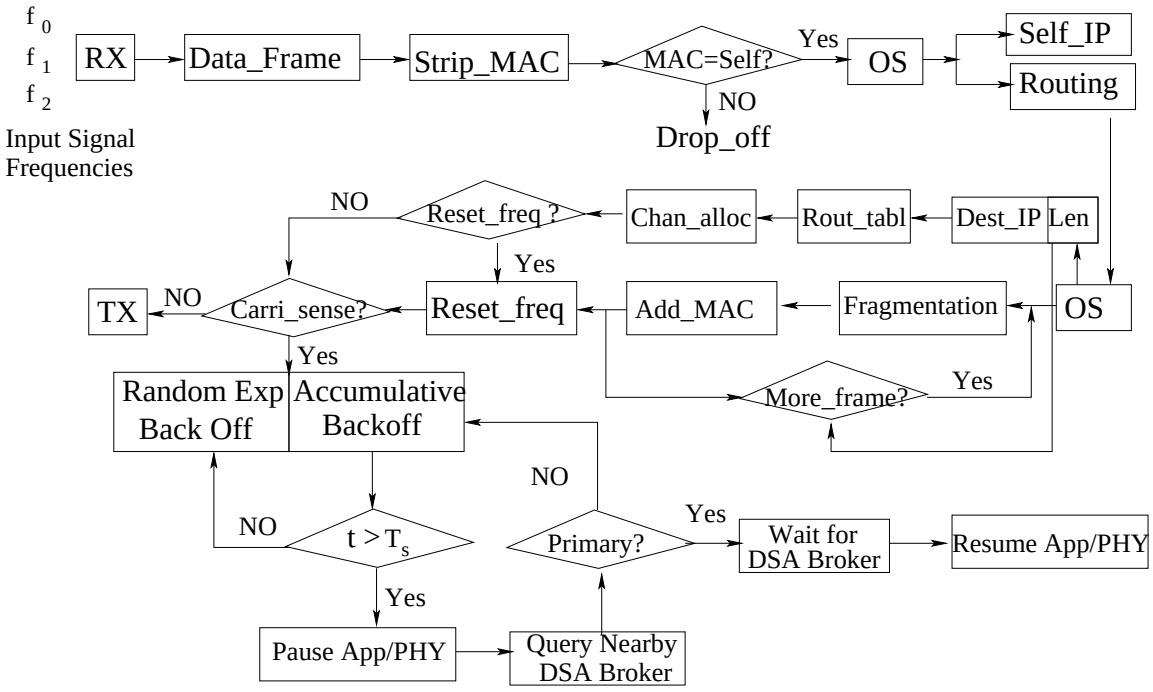


Figure 5.8: MAC layer enabling DSA.

incumbent user is not present, the CR node assumes that its peer nodes are transmitting over the channel. It resets its accumulative time counter to zero and continues its random time backoff before trying its next transmission. If an incumbent user is present, the CR node queries the DSA broker for a new channel. Meanwhile, the DSA broker negotiates with its neighboring DSA brokers and they together determine new channels for their associated CR nodes. Once a CR gets a new channel, it will reconfigure itself to transmit over that channel. It will also re-assume the data communication at its application layer.

Please be aware that signal classification is computationally costly and takes a significant amount of processing time, especially when it covers a wide band [44]. Therefore, it is power inefficient to enable all nodes' signal classification function. A "listen-before-transmit" MAC protocol guarantees non-interference to incumbent users. It also makes it possible that only a small portion of network nodes activate a signal classification function at a given time interval.

However, if incumbent users also use a CSMA based MAC protocol, CR nodes may indirectly interfere with incumbent users because incumbent nodes cannot differentiate their peer nodes' signal from CR nodes'. Therefore, they will back off when a CR node is transmitting on the same channel. To solve this problem, the distributed cooperative spectrum sensor network will (using the control channel) signal CR nodes to switch to other channels once an incumbent signal is recognized.

5.4.3 Network Controller

As shown in Figure 5.6, the network controller works with the learning module and the DSA module to compute appropriate network level protocols and to select suitable parameters in those protocols.

The current network prototype uses an open source implementation of OLSR [135] as its routing protocol. It is a proactive link-state routing protocol which uses Hello and Topology Control (TC) messages to discover and then distribute link state information throughout the ad-hoc network. CR nodes use this topology information to compute next hop destinations for all nodes in the network using shortest hop forwarding paths. With its underlying MultiPoint Relays (MPRs) technique [136], OLSR delivers routing control messages and the IGA's migration information to designated destination nodes with substantial communication cost reduction compared to simple flooding based routing protocols.

5.4.4 Working Mechanism

The previous sections introduce all the functions used in our network prototype; this part details the working mechanism of this network, in particular, supporting distributed cooperative spectrum sensing, multi-channel allocation, and dynamic spectrum access.

Additional Functions and Implementations

Before going further, a particular function is introduced to interface DSA brokers with their associated nodes. Once recognizing an incumbent signal on a channel currently used by CR nodes, a DSA broker must notify its associated CR nodes. In the network prototype, this data communication happens at the network layer. Each DSA module has a particular server agent which continuously awaits data from DSA brokers. The data indicates whether an incumbent user appears or not. Because the SDR system must check this data before it transmits every packet, very importantly, there must not be any delay for such an access. To enable this performance, multi-thread technologies are employed to combine this server agent with the SDR system in one single process.

In another important implementation, the OLSR protocol is modified in this network prototype. This modification allows the control network and the CR data network to use the same routing algorithms, that is; OLSR manages two network interfaces. Because the network topology is the same for these two networks, this implementation enables system simplicity and enhances node performance.

Working Process

In this network prototype, CR nodes opportunistically use vacant channels in their communications when incumbent users are not present. All the nodes are mobile under a dynamic radio environment where fading and interference are normal and incumbent users may appear anytime. The whole network is decentralized and relies on distributed functions and their supporting protocols.

A flowchart is shown in Figure 5.9 to help understand the working mechanism. Before the secondary data network is established, the network connection topology is available to each node through the control channel. Based on this information, the network partitions itself into clusters as shown in Figure 5.1. Each cluster then selects a few nodes and enables their signal detection and classification functions. The selected sensors continuously sense a frequency range and report the results to a selected DSA broker node for data fusion. Upon any radio environment variation, DSA brokers in neighboring clusters exchange their data. All the communications are through the common control channel.

Meanwhile, all the nodes register themselves with the DSA broker in the same cluster. Based on the registration information, DSA brokers derive the local network topology distribution. Next, DSA brokers compute a routing table and a channel allocation table for each node using the distributed multi-channel allocation algorithm discussed in Section 5.3.2. Upon receiving such tables, CR nodes begin to communicate with each other on particular channels.

When incumbent users return, CR nodes sense a busy channel and pause their SDR system and the application. Then they query their closest DSA brokers. At the same time, those DSA brokers will realize the presence of incumbent users through their associated signal recognizers. Next they will select new vacant channels and cooperatively compute a new channel allocation table and maybe a new routing table. Each DSA broker will broadcast the two tables to their associated CR nodes. Finally the CR network re-assumes communications. To reduce power consumption, a DSA broker makes its associated signal recognizers focus only on channels currently used by CR nodes; while it makes its associated signal detectors cover a wide frequency range to find vacant channels for next channel switching once the incumbent users on current channels return.

5.5 Experimental Setup and Results

The experiments intended to investigate the performance of this decentralized DSA network prototype from three aspects: distributed cooperative spectrum sensing, multi-channel

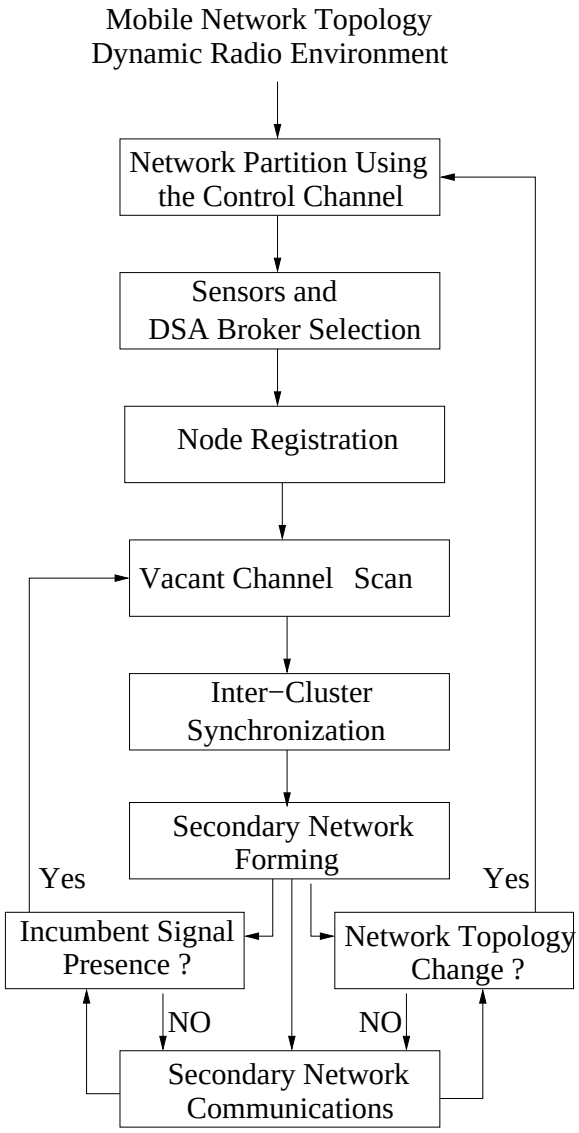


Figure 5.9: The working mechanism in the DSA network prototype.

allocation, and network switching and reforming upon incumbent users' activation.

5.5.1 Experimental Settings

The overall network setting is shown in Figure 5.10. It emulates two clusters of the network shown in Figure 5.1. Each cluster has five nodes including one DSA broker, one signal detector, one signal recognizer with the signal classification algorithm, and two other CR nodes. In particular, one node was allowed only to support one function because of practical limits in dynamic computing resource allocation. Detailed explanations are available in Chapter 2 and [134]. In the experiment, the network was manually partitioned; so were selections of sensors and DSA brokers.

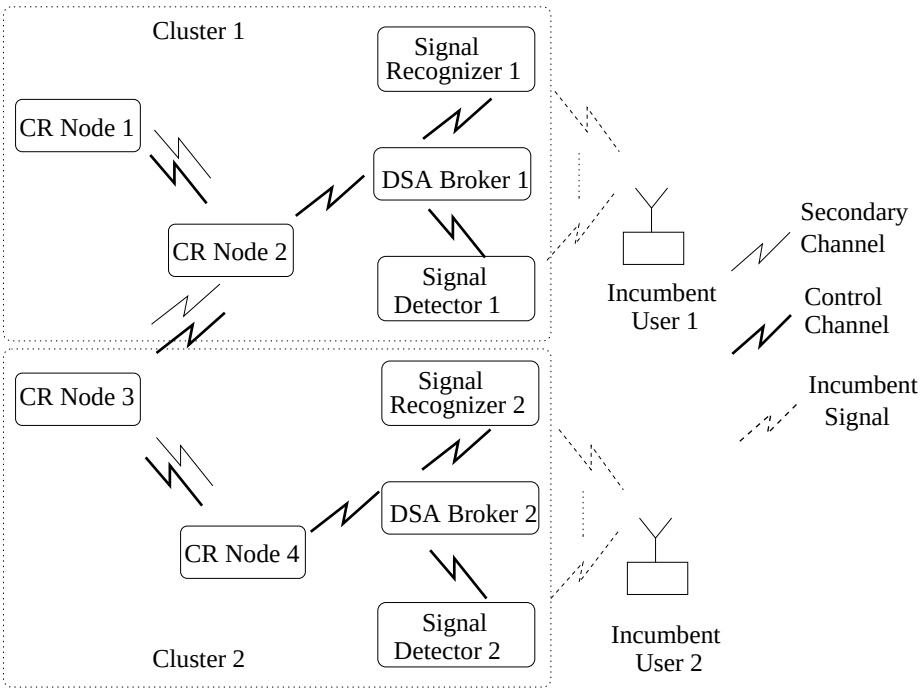


Figure 5.10: The experimental system.

As shown in Figure 5.10, the four CR data nodes form a chain topology for data communication using DSA. Both a signal recognizer and a signal detector associate with each DSA broker. Both DSA brokers talked to each other and other nodes through a Wi-Fi control channel; CR nodes opportunistically used vacant channels in their communications. Two SDR based transmitters were used to emulate primary users. All the nodes consisted of a 2.0 GHz dual-core laptop, a USRP, and a Wi-Fi card .

Because of the limited physical space for this experiment, all the nodes were not separated widely enough. All the Wi-Fi and USRP could sense each other. The above network topology was artificially created by blocking MAC addresses of non-neighbor nodes in Figure 5.10. Therefore, the PHY layer media contention was higher than it in theory. In addition, the experiments were completed in a laboratory environment, exacerbating multi-path effects.

5.5.2 Experimental Scenarios in Radio Environment and Network Variations

The experiment demonstrated three network functions: distributed cooperative spectrum sensing, multi-channel allocation, and dynamic spectrum access. Different experimental scenarios were created for the above demonstrations.

- Distributed cooperative spectrum sensing.
  - Signal Detection and Signal Classification. As shown in Figure 5.10, SDR systems were first used to emulate incumbent users with different transmitting frequencies and modulations, then the author measured the performance of individual sensors regarding signal detection and classification.
  - Distributed Cooperative Spectrum Sensing. Here the author mainly measured the time performance of the distributed cooperative spectrum sensor network – that is, how fast can such a network react to a signal's presence.
- Multi-channel Allocation. The author investigated the performance of the network prototype in multi-channel allocation using the localized IGA algorithm. Performance included network throughput, latency, and jitter.
- Dynamic Spectrum Access. With the results of distributed cooperative spectrum sensing, the author evaluated the time required for channel switching and network reforming when an incumbent signal was present. He measured the latency in channel switching and network reforming as well as data packet loss/data throughput.

5.5.3 Experimental Results

In quantifying the performance of the DSA network prototype, the author followed guidance from [31]: must do no harm; must work; must add value. Specifically, he measured this network's reaction time to active signals, its data communication performance, and its reforming time once an incumbent user appears. Those measurements corresponded to performance in distributed cooperative spectrum sensing, multi-channel allocation, and dynamic spectrum access.

Distributed Cooperative Spectrum Sensing

To investigate the performance of distributed cooperative spectrum sensing, the author varied the location of a sensor in the network topology shown in Figure 5.11. In particular, the sensors were put multiple hops away from the DSA broker. Further, timestamps were inserted at multiple locations to get the timing performance of this distributed cooperative spectrum sensing network at multiple scales. The time symbols used in Table 5.2 are the same as those labeled in this figure.

The quantitative results are shown in Table 5.2. The meaning of each item is explained by the time difference shown in Figure 5.11. Please notice that the execution time of the

Table 5.2: Performance of the distributed cooperative spectrum sensor network.

Node Level						
	Time (ms)					
Trial Number	1	2	3	4	5	Average
Detection Algorithm ($t_3 - t_2$)	21	21	21	22	26	22.2
Classification Algorithm ($t'_3 - t_2$)	681	723	697	745	1053	780
DSA broker Processing ($t_5 - t_4$)	118	74	56	97	127	94
Radio Reconfiguration	7.8	7.1	6.6	8.0	9.6	7.8
Network Level						
Network Awareness ($t_5 - t_1$)						
One Hop	200	185	174	198	163	184
Two Hops	164	234	239	247	254	228
Three Hops	188	228	290	309	293	262
CR Awareness ($t_6 - t_1$)						
One Hop	137	157	262	247	240	208
Two Hops	180	281	283	294	194	246
Three Hops	262	287	295	274	284	281
Network Recognization ($t'_5 - t_1$)						
One Hop (Classification)	898 (601)	926 (683)	878 (726)	965 (737)	1300 (1035)	993 (756)

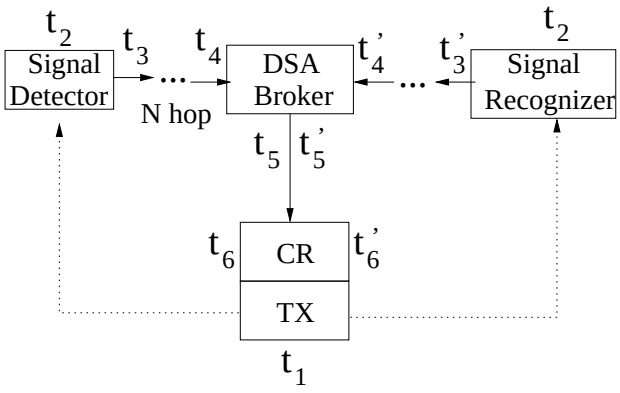


Figure 5.11: Experiment setting to measure sensor network performance.

detection and classification algorithm only accounts for the algorithm running time (for example, energy detection algorithm and local data management for a signal detector). However, a separate measurement indicated that the signal sample acquisition time was about 4 ms, almost negligible. The network awareness time is defined as the overall time duration from when a signal detector first detects a signal to when it updates its closest DSA broker. The network recognization time is similarly defined, except that it is for a signal recognizer, instead of a signal detector. Correspondingly, the CR awareness time includes the network awareness time and also the time used for a DSA broker to notify its associated CR nodes.

As shown in Table 5.2, once signal samples are available, executing the signal detection algorithm takes only about 22 ms while running the signal classification algorithm takes significantly longer, about 800 ms. Although the results depend on the selected hardware and algorithms, such a big time difference illustrates that a signal recognizer's execution time and not only its accuracy will be a critical limiting factor for a DSA network's performance. The other limiting factor at the node level is the time required for data fusion and management in DSA brokers. The results show that it is almost 100 ms on average. Because the signal recognizers focus on channels being used by CR nodes, their reconfiguration time must be counted. The SDR based sensors used here can reconfigure within 10 ms; this is also the performance of the used CR node in reconfiguration.

Also shown in Table 5.2 is the network level performance. The network awareness time performance shows that it takes about 40 ms more for each extra hop. As shown in Figure 5.11, at one hop distance, the network CR awareness seems to include just one more hop than the network awareness; but their performance difference is only about 20 ms, not 40 ms. The reason for this difference is that the DSA broker used a UDP packet to notify its associated nodes, while data communications between a sensor and a DSA broker used TCP packets. Similarly, increasing by one hop in the CR awareness takes

about 40 ms more for the network to react to the presence of an active signal. Given this trend, only the one hop case was measured for the time performance of network level recognization. For this measurement, the signal classification time was also inserted for each measurement. Because signal classification takes more time than signal detection, the network level recognization time is much longer.

Multi-Channel Allocation

In this experiment, the IF band sample rate was set as 1 MS/s, the frequency separation between two adjacent sub-channels was set as 200kHz. Gaussian minimum-shift keying (GMSK) was used at each sub-channel and the data rate was fixed as 50 kb/s. The experiment investigated the performance of multi-channel allocation from four aspects: latency, throughput, jitter, and packet loss.

The ping command was used to measure the network latency in terms of Round Trip Time (RTT). Table 5.3 summarizes the performance results of the multi-channel network and the single channel network at different network distances, in terms of the average RTT, the standard deviation of RTT, and the packet loss percentage . The results show that the RTT of the multi-channel network is much better than the single-channel network, particularly when the network distance increases. The RTT standard deviation comparison between two networks indicates that the multi-channel network has a more stable connectivity than the single channel network. As the network distance increases, the single channel network experiences more packet loss. This is probably due to the higher media contention in this network.

Table 5.3: Multi-channel vs single channel network latency.

	1 hop		2 hops		3 hops	
	Multi	Single	Multi	Single	Multi	Single
Avg RTT (ms)	59.1	67.7	141.8	249.4	203.3	377.6
Std. Dev. (ms)	1.5	3.2	29.1	33.7	23.1	38.1
Packet Loss	0	0	3%	3%	3%	12%

Iperf, a tool to measure the bandwidth and the quality of a network link [137], was used to quantify the network's throughput, jitter, and packet loss. The acquired data results were based on User Datagram Protocol (UDP) and averaged with an interval of 5 seconds over a period 100 seconds. The UDP buffer length was set as 1 kB for reading on the server side and for writing on the client side. Even though the transmitter in the multi-channel network can only support 50 kb/s, the receiver can receive data from 3 subchannels simultaneously,

supporting 150 kb/s. Therefore, the author varied the data bandwidth (the parameter used by the Iperf client to vary the sending data rate) to 5 kB/s, 10 kB/s, and 15 kB/s over both networks. Respectively, Figure 5.12, 5.13, and 5.14 compares the network throughput of the multi-channel network to the single-channel network. The average data throughput of the multi-channel network is 13.8 kb/s and 13.7 kb/s respectively in Figure 5.13 and 5.14. Correspondingly, it is respectively 7.0 kb/s and 7.6 kb/s for the single channel network. Therefore, the multi-channel network can support a data throughput almost twice that of the single channel network when the network data bandwidth is appropriated set. As also shown in Figure 5.13 and 5.14, the multi-channel network experiences less packet loss than the single channel network.

When the data bandwidth is 5 kB/s, however, the single channel network has a better throughput performance (8.0 kb/s on average) than the multi-channel network (7.2 kb/s on average). The multi-channel network also experiences more packet loss than the single channel network. Two complementary reasons may explain this: (1)With a low data bandwidth at the network level, the single channel experiences less contention at the MAC layer, therefore, its data throughput is stable. (2)Because of none perfect filtering to separate the three receiving subchannels in the multi-channel network, some level of adjacent channel interference exists among the three channel. This may cause some percentage of packet loss in the multi-channel network. Since this problem does not exist in the single channel network, its throughput performance is better when there is little media contention. However, as the data bandwidth increases, the single channel network experiences more media contention, its throughput performance deteriorates; but the multi-channel network does not have this problem, its throughput performance maintains at the same level as it for the low data bandwidth case.

Correspondingly, Figure 5.15, 5.16, and 5.17 show the jitter rate of two networks at different data bandwidths at the network layer. The single-channel network almost always experiences a higher jitter rate than the multi-channel network. Further, both networks experience an increase followed by a decrease in the jitter rate. This probably happens because of backoffs and queuing delays at the MAC layer.

Channel Switching and Network Reforming

When a primary signal appears, the DSA network must execute a series of actions dynamically. As shown in Figure 5.10, this experiment was started by letting four CR nodes wait for a vacant channel from their associated DSA brokers. Two signals were then transmitted at different bands to emulate an initial radio environment. They were immediately detected by both signal detectors, which then updated the DSA brokers. Next the DSA bro-

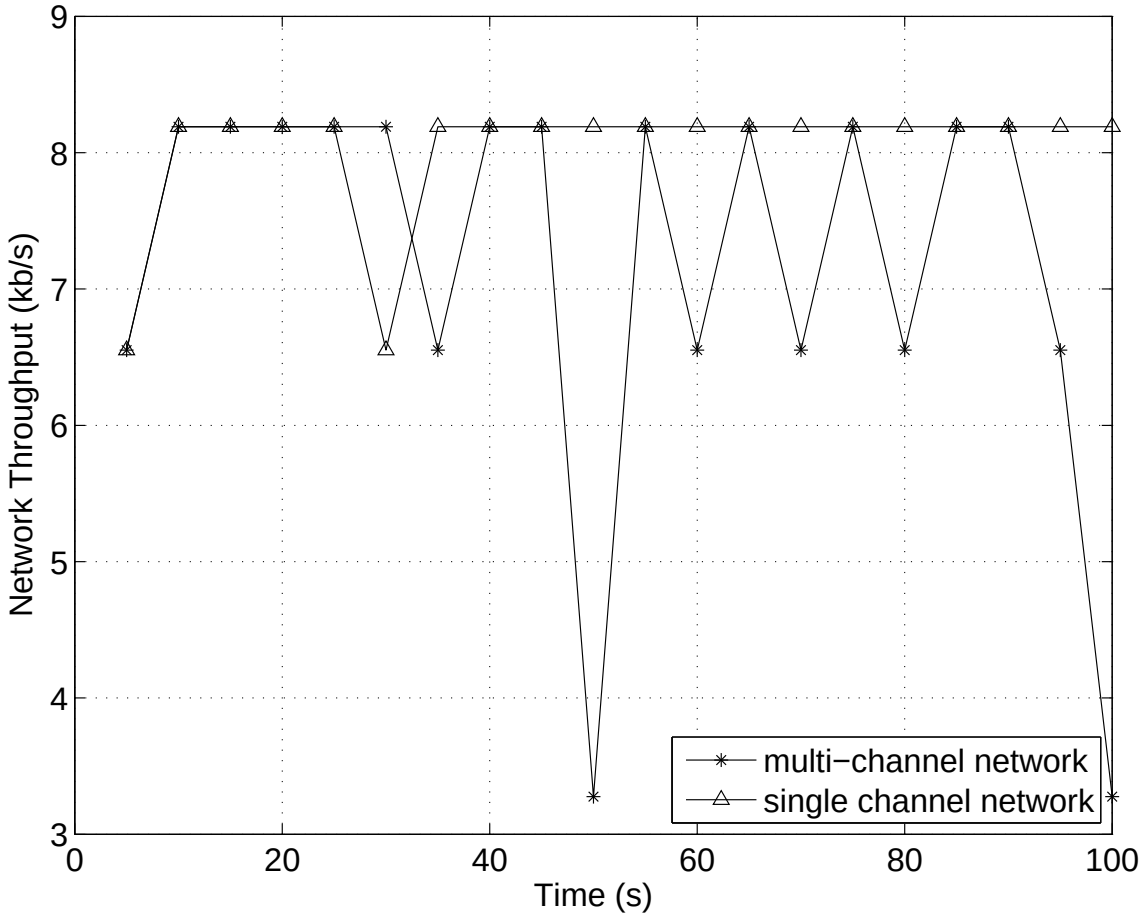


Figure 5.12: Throughput comparison between multi-channel and single channel networks at the Iperf data bandwidth 5 kB/s.

kers calculated the radio environment map and chose a vacant channel for four CR nodes. Meanwhile, the DSA brokers made their associated signal recognizers tune to the center frequency of the chosen channel. Upon receiving channel parameters, four CR node configured and started their SDR subsystems and network controllers. The SDR subsystem used GMSK modulation on its three sub-channels. Finally the CR network established its data communication.

Before the transmission of an in-band primary signal, a pair of Iperf client and server were started to monitor data traffic variation over the network. Next this primary user was emulated by a signal transmission with Differential Binary phase-shift keying (DBPSK) modulation on one of the three subchannels used by four CR nodes. The signal recognizers sensed and classified this new signal. They then sent the signal's parameters to the DSA brokers. The DSA brokers compared this new signal with those stored in their databases, found it to be a primary signal and notified their associated CR nodes to stop transmitting

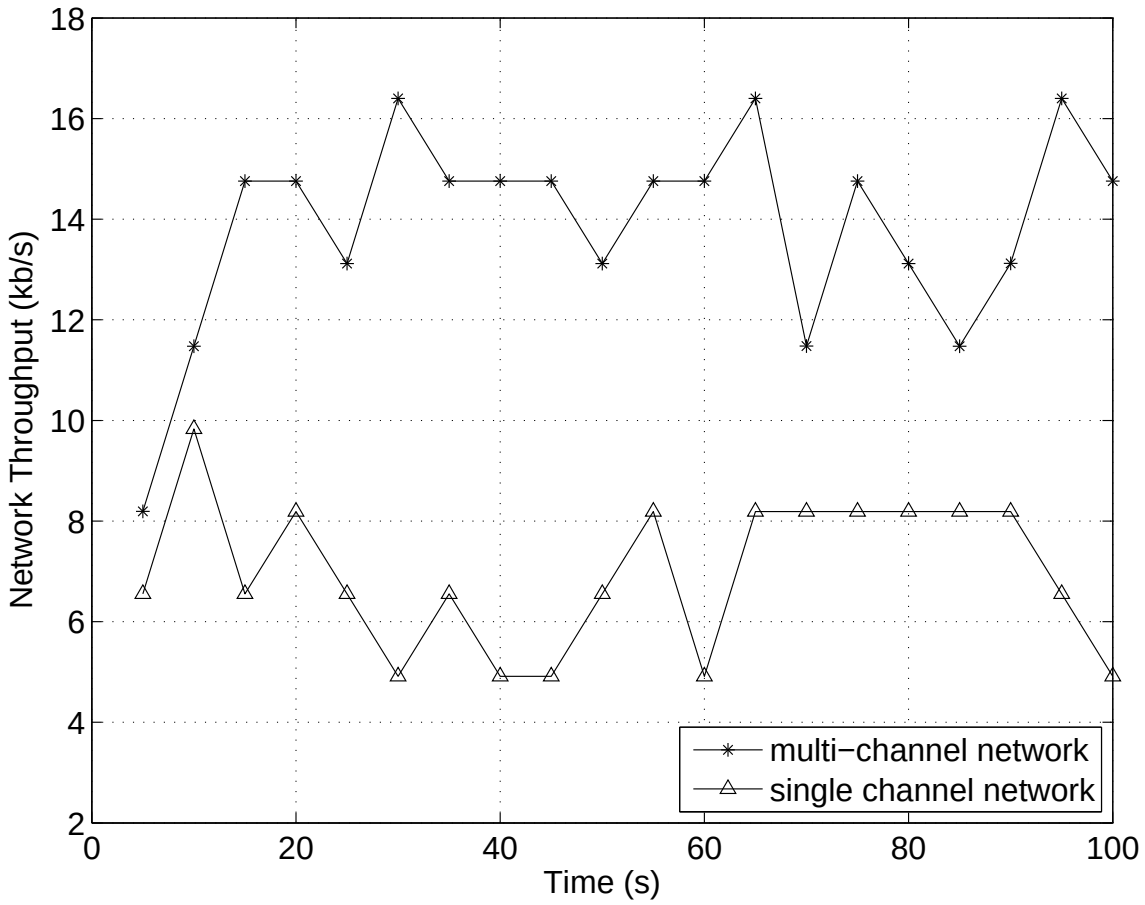


Figure 5.13: Throughput comparison between multi-channel and single channel networks at the Iperf data bandwidth 10 kB/s.

on this subchannel. The CR nodes then reconfigured their SDR subsystems and continued the data communications on the other two subchannels. The running Iperf recorded the network throughput variation during this process.

The above scenarios were repeated multiple times and their performance variation was similar. One case is shown in Figure 5.18, where the primary signal is introduced at 59 seconds. The results are not averaged so as to reveal more details. As shown in Figure 5.18, the network experiences data loss when switching channels. This transition period is about 2 seconds. Also shown in the figure, there is some packet loss at other times. This happens probably because of the multi-path in the experiment environment.

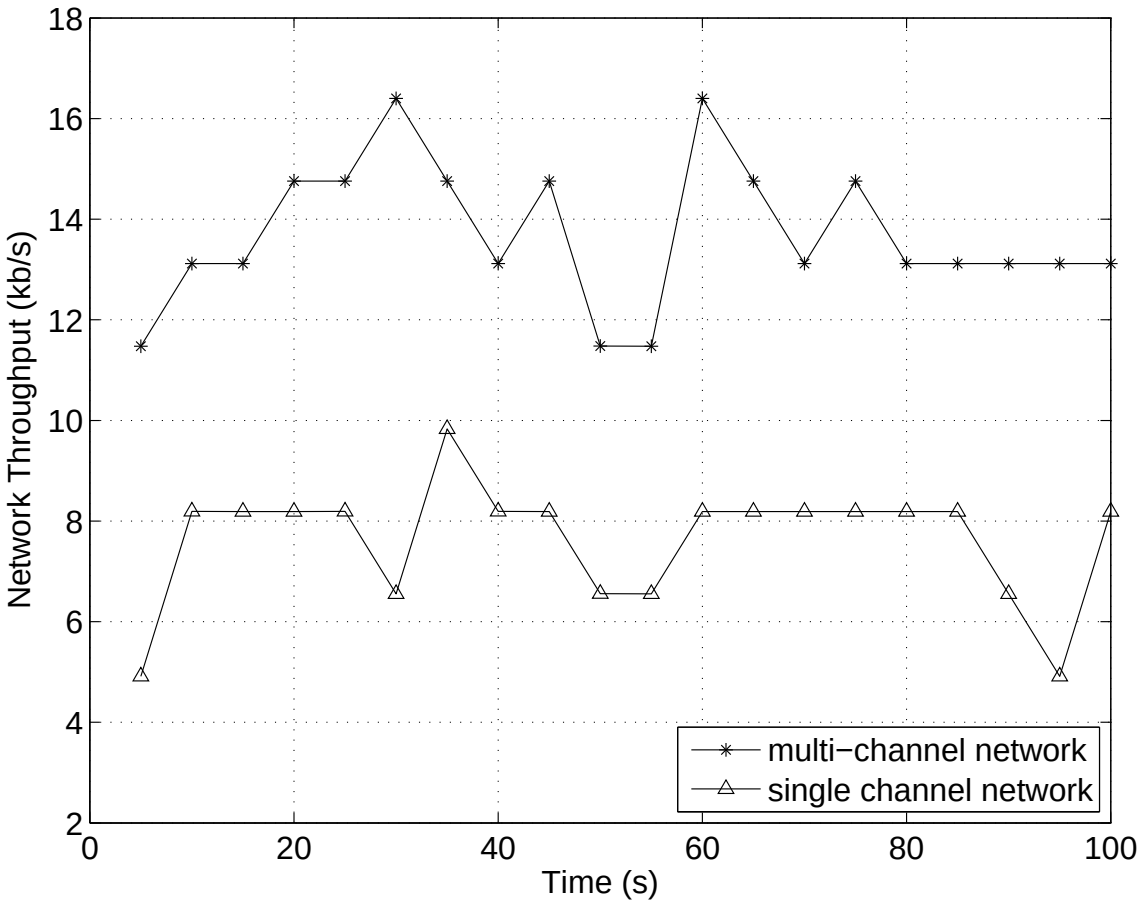


Figure 5.14: Throughput comparison between multi-channel and single channel networks at the Iperf data bandwidth 15 kB/s.

5.6 Lessons Learned

In designing and implementing this network prototype, the author and his colleagues encountered several problems that demanded further investigations. Alternative methods were tried to find the right combination of subsystem design and implementation. The author would like to share our studies and lessons. Some can be generalized to other DSA network design and implementations; others are specific to this particular network prototype and the selected software and hardware.

5.6.1 Dynamic Computing Resource Allocation in SDR

Instead of relying on separate commercial products in building a CR node, (what XG project did [30]), the author used software radio technologies to develop most functions. Initially he followed lessons from the XG project [30] and used multi-threading technology

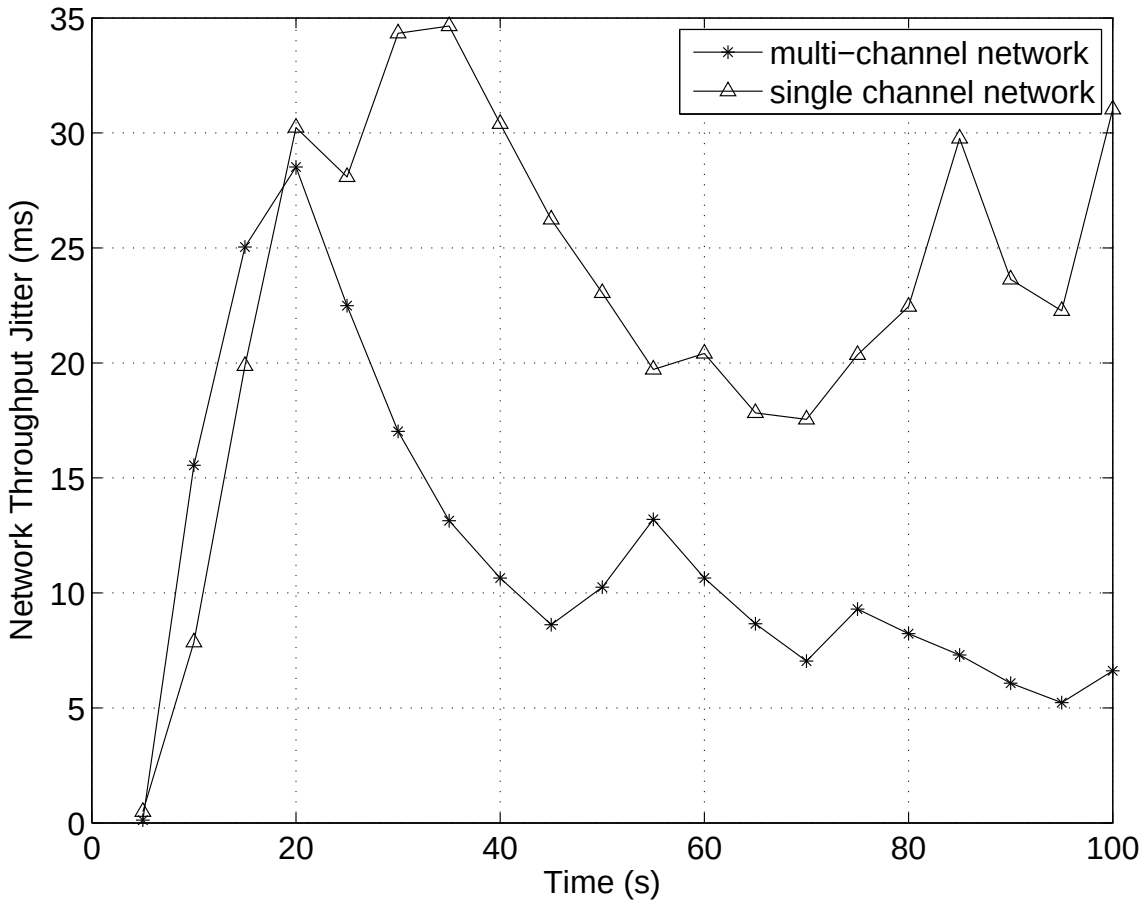


Figure 5.15: Jitter comparison between multi-channel and single channel networks at the Iperf data bandwidth 5 kB/s.

in Python to put in a single program most node functions including signal sensing and and the SDR systems supported by GNU Radio. He thought that it was much easier to manage and control different functions, particularly for quickly passing commands and data among different functions. Unfortunately, the system didn't perform as the author expected, especially the SDR system couldn't transmit and receive data appropriately.

The author later realized that the XG node uses a dedicated digital signal processing board for its communication function (WiMAX) and it also has a standalone sensing unit. However, the author put almost all the functions in a single computing domain. There might be computing resource contention among different function modules. Fundamentally any radio system must guarantee a real-time performance because Data is transmitted as segments at the PHY layer; the time to process each data segment is limited since the radio must accept and process this data segment before the next segment arrives.

To confirm the judgement, the author backed up a step and did another project to inves-

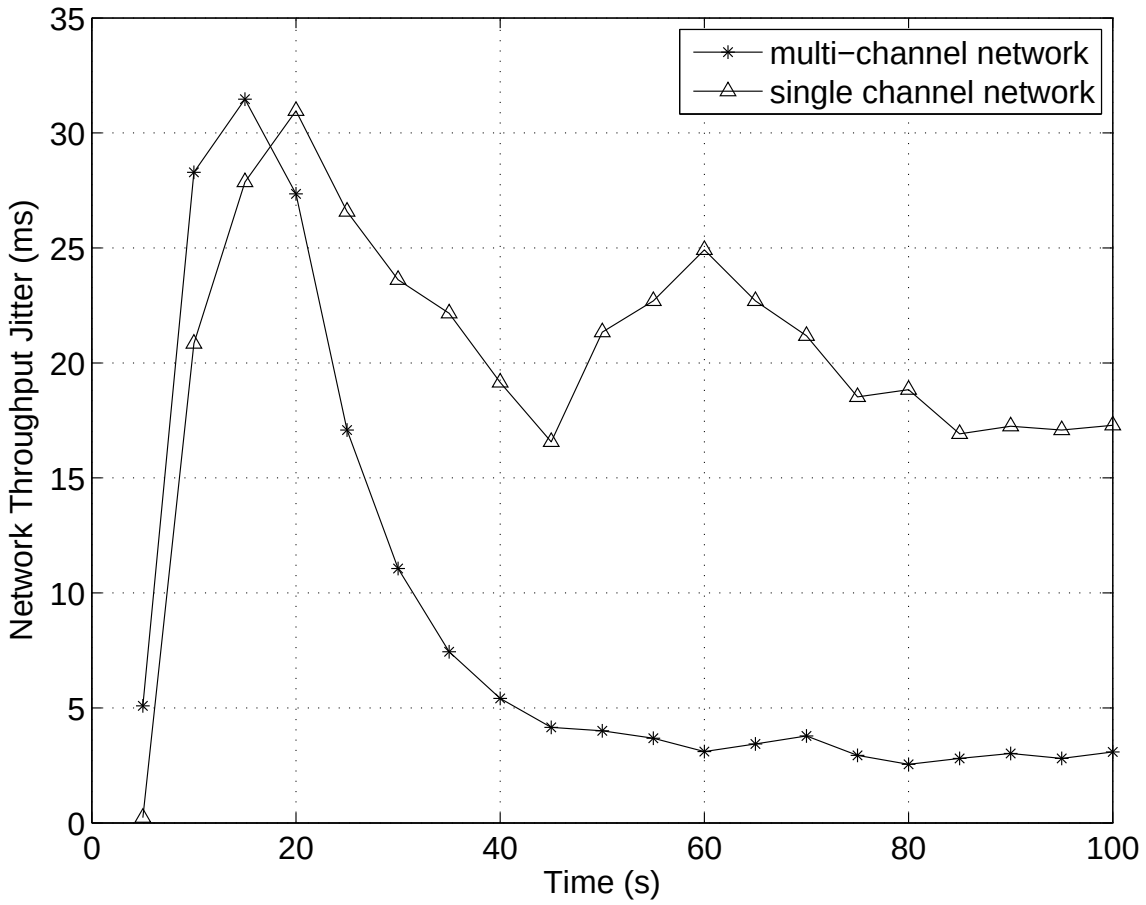


Figure 5.16: Jitter comparison between multi-channel and single channel networks at the Iperf data bandwidth 10 kB/s.

tigate the dynamic computing resource allocation constraint in supporting real-time digital communication functions in general purpose processor (GPP) based SDR. As shown in Chapter 2, further investigations showed that general purpose processor (GPP) based SDR's performance is sensitive to computing resource contention. More details are also available in [134]. Since the SDR system here is mainly developed in a Linux GPP, its functions' execution and their interactions are constrained by the operating mechanism of the operating system (OS). The OS dynamically schedules computing resources in a sequential way to support multiple processes simultaneously, more details are available in [49]. The author concluded that a purely multi-threading method is not suitable to our whole node system. Therefore, he adopted a hybrid process-thread structure to guarantee the SDR system was assigned the highest priority in resource allocation. Other modules were offloaded to special nodes. For example, he did not put the signal detection and classification functions together with the reconfigurable SDR system in a single node, as shown in Figure 5.10.

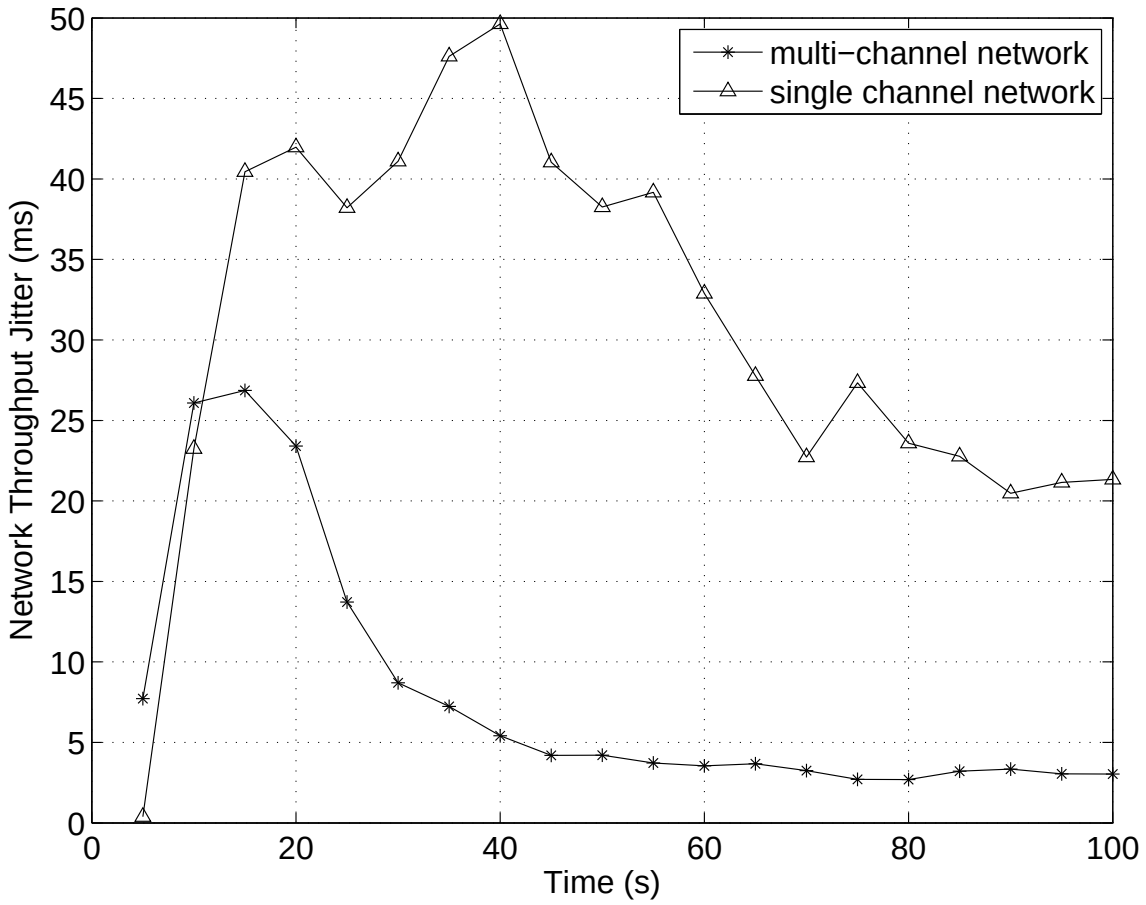


Figure 5.17: Jitter comparison between multi-channel and single channel networks at the Iperf data bandwidth 15 kB/s.

Further, it is challenging to maintain a real-time performance with a high data rate in the software domain in GNU Radio [134]. Therefore, the SDR system here can only support a low data rate at the PHY layer as specified in Section 5.5.3.

5.6.2 Execution Latency in SDR

A wireless network performance is determined by many factors, among them the MAC layer is especially crucial because wireless medium is shared among all nodes and they may interfere with each other. Further, each node may experience shadowing, fading, and multi-path. As a result, a wireless network's PHY layer no longer has a constant channel capacity, in contrast with wired networks [61].

SDR provides remarkable flexibility by moving to the software domain a lot of functions that are conventionally supported either in the analog domain or in ASICs. As de-

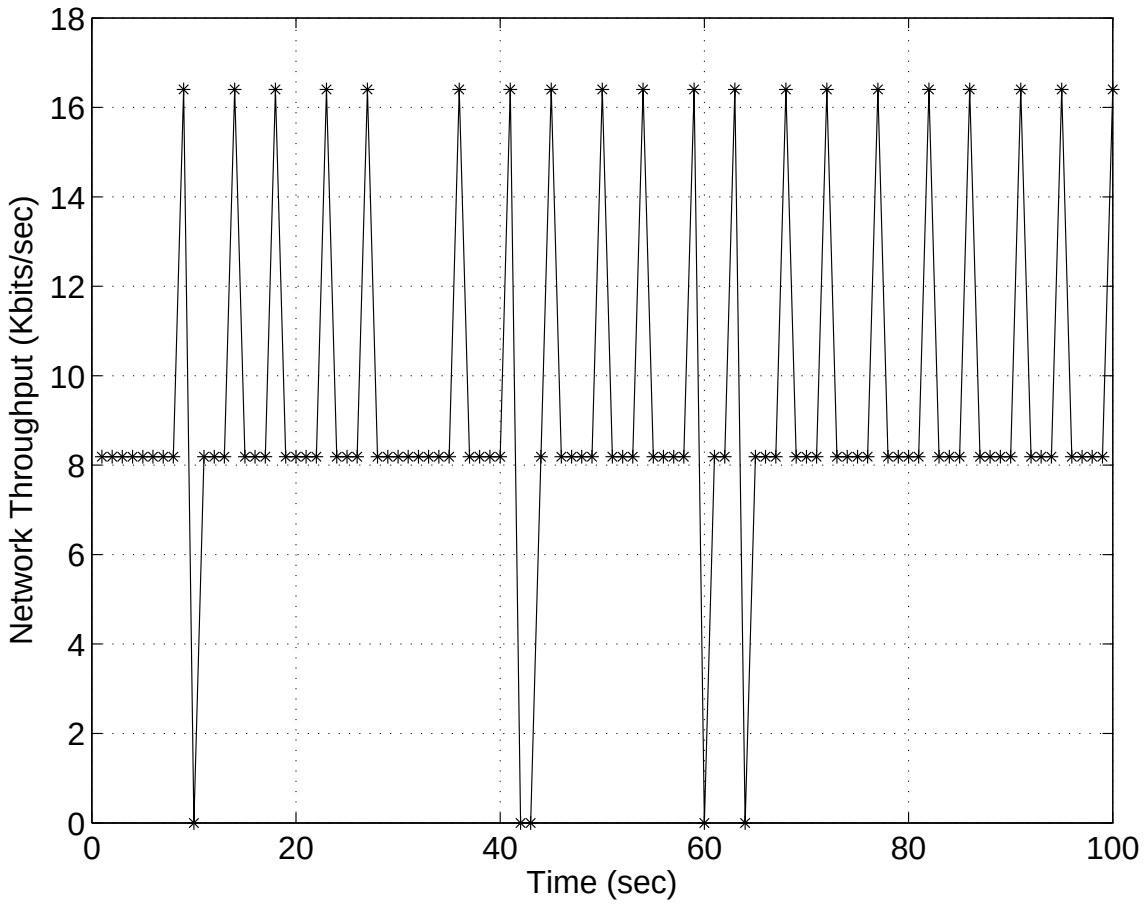


Figure 5.18: Throughput variation in dynamic spectrum access. The primary user began its transmission at 59 seconds.

tailed in Chapter 2, however, this deteriorates GPP-based SDR's timing performance in supporting MAC layer functions because of delays in the memory hierarchy in the GPP, the peripherals connecting the RF front end to the GPP, and multiple data buffers along the radio transceiver signal processing chain as well as resource contention in the OS. Results from Chapter 2 indicate that the time taken between receiving and transmitting a packet in GNU Radio is at the order of 10 ms. However, the MAC protocols used in Wi-Fi require a timing performance at the order of 10 μs. The author concluded that Carrier Sense Multiple Access (CSMA)-based MAC protocols would lead to poor performance in our system, which is verified by our experimental results shown in Figure 5.12 – 5.18.

5.6.3 Others

The author also experienced severe problems because of software and hardware incompatibility. For example, the signal classification algorithm was written in Matlab code, while the signal sample acquisition was through Python based GNU Radio and the DSA broker was written in C++. Because Matlab and Python code have subtle differences for specific functions, the author had some difficulty in properly sorting out their interactions. Moreover, because of resource limitation, the author used Wi-Fi chips from different manufacturers. This often made the OLSR break a whole ad hoc network into several small segments; communications through the whole network therefore failed frequently. Overall, compatible software and hardware can save a lot of time and achieve better performance in large system development than incompatible ones.

5.7 Conclusion

This chapter describes the complete design of a software radio-based decentralized DSA network to support distributed cooperative spectrum sensing, multi-channel allocation, and dynamic spectrum access. Further, it introduces this network prototype's enabling technologies including a software radio system reconfigurable at the PHY layer and MAC layer, signal detection and classification methods, a distributed cooperative spectrum sensing system, a modified MANET protocol, and a distributed learning algorithm for channel allocation. Performance results of a ten-node network prototype are presented under three scenarios: cooperative spectrum sensing, multi-channel allocation, and dynamic spectrum access. Such results reveal several factors determining performance for developing decentralized DSA networks.

However, the current network prototype was built mainly from off-the-shelf hardware, therefore, its performance is limited by the hardware. For example, running the signal classification algorithm on a GPP takes about one second. Nonetheless, the experimental results acquired from a real network do reflect important facts.

First, function distribution must consider the tradeoff between node level resource consumption and network level communication cost. Dispersing functions over the network can reduce the demand on energy and computing resources in each node; but it is associated with time delay and communication overhead. Therefore, it must be guaranteed that such associated cost is within a needed performance bound. More investigations are needed in decentralized CR networks which are both time and performance critical.

Second, even with a high-bandwidth control channel in the current network prototype,

its distributed cooperative spectrum sensing system experiences a significant time delay when the network distance is three hops. However, a ping message takes only about 8 ms on average. Given that the communication happened at the network layer and TCP/IP packets contain big headers, special packet formats are needed for sensor data exchange. Even further, it might be possible to enable some of such data exchange at the MAC layer.

Third, decentralized DSA networks experience connection outages when in-band primary signals are detected. Because DSA nodes must immediately stop their transmission on that channel, data packets held in intermediate nodes along data routes may be lost once the network connection is reestablished. This problem is particularly severe when the network is at a large scale. Some disruption tolerant protocols can mitigate this problem.

Chapter 6

Conclusions

Supporting DSA in decentralized wireless networks drives the complexity both within individual nodes and over a whole network to an unprecedented level. Experimental methods, besides theoretical analyses and computational simulations, are necessary to gain insights into design of such networks and to investigate their performance under different scenarios. However, enabling DSA in such networks requires innovative protocols, advanced sensing methods, novel node architecture, and flexible radio platforms, all with a robust performance. This dissertation has taken a bottom-up approach to designing a decentralized and asynchronous DSA network and implementing a ten-node prototype. Further, this dissertation has presented research on several enabling technologies: software radio, signal detection and classification, distributed cooperative spectrum sensing, and reconfigurable node architecture. Finally, the dissertation has identified several performance determining factors for decentralized DSA networks.

More specifically, Chapter 1 gave a brief background on dynamic spectrum access and a short review on the evolution of radio technology, arguing that now is the time to realize DSA. This chapter then surveyed the current research efforts in DSA technology, targeting both centralized and decentralized networks. Chapter 2 investigated three practical issues governing GPP-based SDR performance: RF front end nonlinearity, dynamic computing resource allocation, and execution latency. This chapter also gave quantitative results on SDR performance. It concluded that great care must be paid when applying GPP-based SDR in CR and DSA network development, and that unavoidable performance limitations exist. Chapter 3 reviewed different methods in signal detection and classification, as used in DSA networks to guarantee noninterference to primary users. It then focused on developing a parallel-computing-based spectrum sensing approach for signal detection under conditions of low SNR and Rayleigh multi-path fading [44]. The chapter further presented the method's performance on multiple signal types under noisy and multi-path fading en-

vironments. Chapter 4 first went over research progress in decentralized detection from a hypothesis test perspective. It then explored performance issues in cooperative spectrum sensing in DSA networks – requiring low data latency and high data reliability. Next it provided a set of schemes for spectrum data flow and process management within spectrum sensors and DSA brokers. The chapter also built a cooperative spectrum sensing system that gives a complete semantic map of the radio environment. Finally, Chapter 5 first described the design and implementation of a decentralized and asynchronous DSA network prototype, and its constituting functions and components. It then presented the performance results of this network prototype in cooperative spectrum sensing, multi-channel allocation, and dynamic spectrum access.

6.1 Future Research Discussion

This section describes the author's further thoughts on dynamic spectrum access and the evolution of radio technology. In particular, the author wonders about how much intelligence a radio can possess and the research challenges in networks composed of highly adaptive or intelligent nodes.

6.1.1 Spectrum Resource Usage Optimization

As exemplified by smart-phones, broadband wireless devices are multiplying quickly. Furthermore, the total data load through mobile devices is increasing exponentially. For example, Cisco's visual networking index forecasts that the global mobile data traffic will nearly double every year from 6 petabytes per month in 2008 to 397 petabytes per month in 2013 [9]. On the other hand, the allocated *sweet* spectrum for broadband communications is limited. Moreover, current policy makes spectrum reallocation difficult and always many years behind the market need. Therefore, dynamic spectrum access, as a new paradigm to both increase spectrum usage efficiency and offer easy spectrum access to new service providers, will likely keep attracting attention from policy makers, the market, and the research community. However, even though applying DSA is now technically feasible, some stakeholders (who have licensed large chunks of spectrum) won't give up their opposition easily. Therefore, it's hard to predict exactly when or even whether DSA technology will be widely applied.

If we only consider spectrum utilization efficiency, DSA argues from the perspective of opportunistically using vacant spectrum at specific bands within a particular geographic region and at a particular time. Conceptually, DSA intends to best use spectrum resources in

three dimensions: frequency, time, and space. However, the involved *space* is in fact either two dimensional (for example, terrestrial wireless communications), or three dimensional (for example, military communications require an integration of air, space, and ground communications). In the U.S, there exist a vast number of wireless devices. At any moment, some of them may be talking to each other and they may cause interfere to their neighbors (in domains of geography or frequency), while others are either idle or separated far from their neighbors. Therefore, the problem of spectrum utilization optimization is a great-number-of-variable optimization problem in a four or five-dimensional space. Even if we only consider this problem for some specified band, the number of wireless devices may be still huge. If we index each wireless device as a graph node, we have to form edges (maybe directed and weighted) between different nodes to indicate their interaction (both communications and interference). Wireless devices may be active/inactive or mobile/static at any time, and they may establish communications with different devices at different times. Therefore, optimizing spectrum allocation requires solving simultaneously many coupled-optimization problems (for maximizing communications and minimizing interference over different connected networks) over the graph. The computational requirement might be enormous, and it must finish quickly because wireless communications and associated interference happen almost instantly.

However, efficient algorithms might be available because: (1) wireless devices won't be active all the time; (2) the number of nodes in a communication network may be small; (3) the interference region (both in geography and frequency) of each wireless device might be limited. Therefore, the above graph might be sparse. In extreme cases, there is plenty of spectrum but a small number of wireless devices and no spectrum contention or interference exists; therefore, no optimization algorithm is needed. But this is far from the situation that we are facing today. The overcrowding of the ISM and cellular bands indicates that their abstraction graphs are not sparse at all.

The above analysis makes several assumptions: (1) All the information (the geo-locations, the activities and connection of wireless devices) is available. (2) There exists a *super* agent that can instantly (or within a short time) get the above information and make decisions based on some optimization algorithm. (3) The optimization results can reach individual wireless devices immediately (or bounded by a small time delay). In reality, all these assumptions cannot be satisfied because instantly acquiring and delivering information needs a ubiquitous infrastructure with a high bandwidth. Further, a tremendously powerful computer might be needed by the *super* agent to store and process the vast amount of data in the optimization problem. That is, the cost of information acquisition and management to achieve the optimum spectrum resource utilization makes itself infeasible. Decentralized

systems, which don't use global information described here, are practical solutions. But challenges exist too, as Section 6.1.3 will discuss.

6.1.2 The Evolution of Radio Technology

As the demand of broadband services increase continuously, radio technology will likely continue exploiting efficient ways of using a wider range of frequencies with complex waveform and coding, to support higher data rates. For example, MIMO-based CDMA or OFDMA are dominating next generation wireless standards.

However, another trend is also becoming clear – radios, at least in many applications, will become more software based, more flexible, and even adaptive and intelligent, instead of hardware based with pre-determined and fixed functions. This is indicated by a broad range of research and development in software defined and cognitive radio.

In particular, the intersection of the above two trends will bring a lot of opportunities and also big challenges. For example, there are a large number of variables that can be adjusted by MIMO based OFDM radios to achieve good performance under dynamic channel environments characterized by interference, multi-path, and fading. A high level of reconfiguration can allow such radios to adapt to different environments, while achieving a good performance with efficient resource consumption. However, enabling this level of reconfiguration is quite a challenge for design and implementations of both radio platforms and computing devices. Further, efficient algorithms are needed to assess arbitrary situations quickly and find good or optimal solutions to reconfigure the radio platform and achieve good performance.

Even greater challenges come into view if we consider wireless networks composed of cognitive radios, particularly large scale decentralized networks. Conventionally, research in wired Internet considers the PHY layer as a simple model – providing a finite (almost fixed) amount of data rate for the higher layers. Each node is modeled as an autonomous system; so is the whole network. As wireless networks become ubiquitous, the story changes a lot. The wireless PHY layer no longer has a constant channel capacity due to shadowing, fading, multi-path and interference [61]. The variation of channel capacity makes it challenging to maintain acceptable quality of service (QoS). Cross-layer design has been widely studied to overcome this challenge [34]. Thus, the autonomous system model becomes more complicated because it must be aware of dynamics in both radio and network environments. Still, each node can be modeled as a finite state machine (FSM) and its behavior is predefined and deterministic.

However, if the above wireless node is substituted with cognitive radio, research meth-

ods used to study Internet and conventional wireless networks are not anymore sufficient. Besides situation awareness, a CR can make decisions about its operating behavior. However, the decision process is not fully predefined because a CR can learn and adapt. Therefore, FSM-based models, if used to study CR or CR networks, may become completely intractable. Probability based models can reduce the complexity; but such models may not be able to fully capture a CR's behavior [138].

The closest model that we can use is *a learning agent*, a general model of which is shown in Figure 6.1. But this is just the first step to model a cognitive radio network and it does not solve the problem. In fact, theories, methods, and tools are not yet fully available to study such a problem, where multiple learning agents interact with each other in a dynamic environment, perhaps some trying to maximize their individual performance (for CR networks, e.g., power consumption and available bandwidth), while others cooperatively working towards some goals (for CR networks, e.g., communications and resource consumption optimization). Game theory has been applied to study CR networks on particular problems. Though game theory in general has achieved a lot of success, it is focused primarily on the outcome of games (for example, the Nash Equilibrium). CR networks (particularly a heterogeneous network of CR networks), might behave mostly within some regions far away from an equilibrium point (for example, each node or network operates in a suboptimal way), because the huge dynamics in user requirements and radio/network environments does not allow the CR network to evolve a long enough time to reach an equilibrium. That is, the behavior and dynamics of a CR network, not just the possible equilibrium points to be reached in interactions among its comprising entities, demand thorough exploration.

6.1.3 Complex Systems, Artificial Intelligence, and Cognitive Radio (Networks)

Section 6.1.1 indicates that the optimal spectrum utilization is impossible for a huge radio ecosystem to achieve. The term *diminishing returns* reminds us that often we should not go to extreme in optimization without being aware of the associated cost. Therefore, information sharing and control in possible future CR networks (or a heterogeneous network of CR networks) must be decentralized, perhaps with some variations in degree and extent. Such a network, as discussed in Section 6.1.1, will not surrender itself to research methods that decompose it to sub-parts (or individual nodes) and then study it merely from its component parts. Similar systems widely exist, for example, the ecosystem, the human society (e.g., the stock market), and the US power grid. They are characterized by adaption, self-

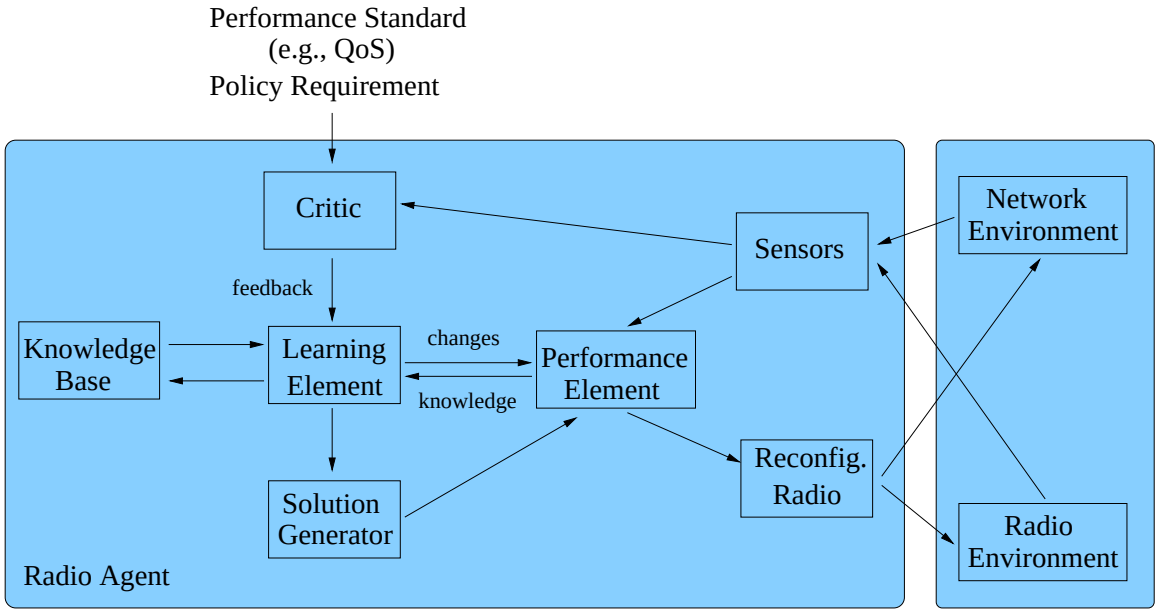


Figure 6.1: A general agent model of cognitive radio.

organization, and emergence [139]. The Nobel laureate Philip Warren Anderson describes such systems (more precisely, in physics and chemistry) as *more is different* [140]. Another Nobel laureate Robert B. Laughlin questions the validity of reductionism to study them and proposed *the middle way* [141]. Despite significant recent advances in our understanding of complex systems, general theories and tools are yet to be developed [139].

In particular, the author is deeply fascinated by the human brain, a complex system understanding which, in his view, will be the summit of all the human research. Real brains have a vast connectivity of nerve cells, the biological equivalent of computer signaling switches. Nerve cells typically form tens of thousands of connections with their neighbors. These features allow real brains to process multiple streams of information in parallel. In contrast, traditional computer switches typically possess only two or three connections with their neighbors, and the basic PC can only process steams of information one step at a time. Without a deep understanding of how real brains work, some experts believe, artificial intelligence (AI) will limit its success in specific narrow domains, for example, playing chess [142].

Several government agencies echoed this opinion. The U.S. National Academy of Engineering, requested by the U.S. National Science Foundation, convened a panel consisting of a diverse committee of experts from around the world. In 2008, this panel proposed the 14 *Grand Challenges*. Among them is *reverse-engineer the brain* [142]. The panel states “reverse-engineering the brain, to determine how it performs its magic, should offer

the dual benefits of helping treating diseases while providing clues for new approaches to computerized artificial intelligence". To call the society's attention to those opportunities and challenges, an inaugural summit was held in March 2009 and a series of summits are planned for 2010.

Regarding *reverse-engineer the brain*, the Defense Advanced Research Projects Agency (DARPA) has made a bold move. In 2008, it started a research program – Systems of Neuromorphic Adaptive Plastic Scalable Electronics (SyNAPSE), which intends to "create new electronics hardware and architecture that can understand, adapt and respond to an informative environment in ways that extend traditional computation to include fundamentally different capabilities found in biological brains" (quoting DARPA program manager Todd Hylton). As the dissertation is being written, one research group from IBM has made some significant progress in simulation. This group has performed the first near real-time cortical simulation of the brain that exceeds the scale of a cat cortex and contains 1 billion spiking neurons and 10 trillion individual learning synapse [143].

Reverse-engineering the brain is relevant, in three-fold, to cognitive radio (network) research. First, the mainstream research now to make a radio *intelligent* is applying existing AI or machine learning methods to the radio domain, people (including the author) may question how intelligent CR can become following this direction. As stated above, domain experts have already began to question methods used in the past AI research; in particular, little attention has been paid to real brains. Such AI also suffers from a fundamental flaw in that it fails to adequately address what intelligence is or what it means to understand something. In fact, even Turing himself could not define what is intelligence, instead, he gave the Turing Test as an answer – which is not a scientific definition. The central dogma in the past AI research has been that *the brain* is just another kind of computer. However, as indicated above, real brains are different from computers.

Second, if we adopt a different way to make radio intelligent, how much success can we achieve? Some applications using AI have benefited from simulations based on brain reverse-engineering. Examples include AI algorithms used in speech recognition and in machine vision systems in automated factories. However, both speech recognition and machine vision have a biological origin and focus on specific domains. In contrast, radio communications, though following fundamental principles like Maxwell's equations and Shannon's theorem, primarily work in vastly dynamic contexts (thinking about the dynamics in the radio and network environment and so many different applications). Therefore, cognitive radio might need to possess some general AI ability to handle its situations, while general AI is still an open research question faced in the field. Further, if new research efforts like SyNAPSE make breakthroughs, new hardware architecture will emerge. How

should we integrate such architecture into ubiquitous mobile devices targeting at different applications? Once the integration is achieved, the AI core might manage both communications and different applications (some are yet to appear), as well as other tasks like resource management.

Third, real brains contain a highly complex and dynamic network composed of hundreds of billions of neurons, methods and tools applied to study this network can possibly be applied to studying the future radio ecosystem – many heterogeneous wireless networks composed of a large number of intelligent nodes supporting various applications and working under dynamic environments.

6.1.4 Contributions and Future Work

In summary, this dissertation has taken a bottom-up approach to designing and buiding a software radio based decentralized DSA network prototype. Experimental results of this prototype identified several performance determining factors for decentralized DSA networks. Further, this dissertation developed enabling technologies for the network prototype including signal detection and classification, distributed cooperative spectrum sensing, and reconfigurable DSA nodes. Moreover, it also investigated GPP-based SDR's performance issues.

As more intelligence is introduced into decentralized DSA networks and their size increases, not only their performance, but also their behaviors and dynamics demand further exploration. But our current network prototype contains limited "intelligence" and operates at a small scale. Our next plan is to integrate more cognitive capabilities like those used in [22] and to scale up the network prototype to about 40 nodes.

Our experiments focused at the functional perspective of the network prototype. However, we did experience a lot of system and network unreliability. Considering the critical importance of reliability in DSA networks, we are planning further explorations of this aspect in decentralized DSA networks.

Further, our current network prototype relies on a control channel. Using a common control channel may raise several questions, for example, who should manage the control channel, what is the right bandwidth of the control channel, should the control channel be fixed at a specific frequency or can it be dynamic too? Rendezvous protocols can allow cognitive radios to talk to each other without a common control channel [144, 145]. Our future work will apply rendezvous protocols in the network prototype.

Bibliography

[1] I. Tolstoy, *James Clerk Maxwell : a biography*. Chicago : University of Chicago Press, 1982.

[2] T. H. Lee, *The Design of CMOS Radio-Frequency Integrated Circuits*. Cambridge University Press, second edition, 2004.

[3] J. Pereira, "Radio resource management in a heterogeneous, converged environment - towards higher spectral efficiency," in *International Union of Radio Science (URSI) proceeding*, New Delhi, India, 2005.

[4] M. A. McHenry, P. A. Tenhula, D. McCloskey, D. A. Roberson, and C. S. Hood, "Chicago spectrum occupancy measurements & analysis and a long-term studies proposal," in *Proceedings of the first international workshop on technology and policy for accessing spectrum*. New York, NY, USA: ACM, 2006, p. 1.

[5] D. A. Roberson, C. S. Hood, J. L. LoCicero, and J. T. MacDonald, "Spectral occupancy and interference studies in support of cognitive radio technology deployment," in *1st IEEE Workshop on Networking Technologies for Software Defined Radio Networks*, Sept. 2006, pp. 26–35.

[6] R. Chiang, G. Rowe, and K. Sowerby, "A quantitative analysis of spectral occupancy measurements for cognitive radio," in *IEEE 65th Vehicular Technology Conference*, April 2007, pp. 3016–3020.

[7] "Spectrum occupancy measurements," Tech. Rep. [Online]. Available: http://www.sharedspectrum.com/measurements/

[8] "More than four billion mobile phone users worldwide, source european information technology observatory (EITO)," August 2009. [Online]. Available: http://www.eito.com/pressinformation_20090807.htm

[9] "Cisco visual networking index: Global mobile data traffic forecast update," 2009. [Online]. Available: http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.html

[10] G. Staple and K. Werbach, "The end of spectrum scarcity [spectrum allocation and utilization]," *IEEE Spectrum*, vol. 41, no. 3, pp. 48–52, March 2004.

[11] J. Chapin and W. Lehr, "Cognitive radios for dynamic spectrum access - the path to market success for dynamic spectrum access technology," in *In Proc. of the IEEE New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2007.

[12] J. K. Hargreaves, *The Upper Atmosphere and Solar-Terrestrial Relations*. Cambridge University Press, 1992.

[13] J. Proakis, *Digital Communications*. McGraw-Hill Science/Engineering/Math; 4 edition, 2000.

[14] J. Mitola, *Software Radio Architecture: Object-oriented Approaches To Wireless Systems Engineering*. Wiley-Interscience, 2000.

[15] "IEEE standard definitions and concepts for dynamic spectrum access: Terminology relating to emerging wireless networks, system functionality, and spectrum management," *IEEE Std 1900.1-2008*, pp. c1–48, 26 2008.

[16] J. H. Reed, *Software Radio: A Modern Approach to Radio Engineering*. Englewood Cliffs, NJ: Prentice-Hall, 2004.

[17] J. Mitola, "Cognitive radio: An integrated agent architecture for software defined radio," Ph.D. dissertation, Royal Institute of Technology (KTH), Sweden, 2000.

[18] P. Marshall, "Extending the reach of cognitive radio," *Proceedings of the IEEE*, vol. 97, no. 4, pp. 612–625, April 2009.

[19] A. MacKenzie, J. Reed, P. Athanas, C. Bostian, R. Buehrer, L. DaSilva, S. Ellingson, Y. Hou, M. Hsiao, J.-M. Park, C. Patterson, S. Raman, and C. da Silva, "Cognitive radio and networking research at Virginia Tech," *Proceedings of the IEEE*, vol. 97, no. 4, pp. 660–688, April 2009.

[20] T. W. Rondeau and C. W. Bostian, *Artificial Intelligence in Wireless Communications*. Artech House Publishers, 2009.

[21] T. R. Newman, "Multiple objective fitness functions for cognitive radio adaptation," Ph.D. dissertation, University of Kansas, 2008.

[22] F. Ge, Q. Chen, Y. Wang, T. W. Rondeau, B. Le, and C. W. Bostian, "Cognitive radio: From spectrum sharing to adaptive learning and reconfiguration," in *IEEE Aerospace Conference*, Big Sky, Montana, 2008.

[23] "Broadband gaps document (doc-294708a1)," Federal Communications Commission, Tech. Rep., 2009.

[24] K. Negus and A. Petrick, "History of wireless local area networks (WLANs) in the unlicensed bands," *info*, vol. VOL. 11 NO. 5, pp. 35–56, 2009.

[25] B. Sham, "First white spaces network brings broadband internet to rural america over unused tv broadcast airwaves," October 21 2009. [Online]. Available: http://www.businesswire.com/portal/site/home/permalink/?ndmViewId=news_view&newsId=20091021005394&newsLang=en

[26] "Part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications: Spectrum and transmit power management extensions in the 5 ghz band in europe, IEEE 802.11h, supplement to IEEE 802.11 standard-1999 edition," IEEE, Tech. Rep., 2003.

[27] "IEEE standard for information technology - telecommunications and information exchange between systems - local and metropolitan area networks - specific requirements part 11: Wireless lan medium access control (MAC) and physical layer (PHY) specifications amendment 3: 3650-3700 mhz operation in usa," Tech. Rep., June 2008. [Online]. Available: http://hraunfoss.fcc.gov/edocs_public/attachmatch/FCC-07-99A1.pdf

[28] "IEEE standard for local and metropolitan area networks part 16: Air interface for fixed broadband wireless access systems," IEEE, Tech. Rep., 2004.

[29] C. Cordeiro, K. Challapali, D. Birru, and N. Sai Shankar, "IEEE 802.22: the first worldwide wireless standard based on cognitive radios," in *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks*, Nov. 2005, pp. 328–337.

[30] M. McHenry, K. Steadman, A. E. Leu, and E. Melick, "XG dsa radio system," in *IEEE Symposium on New Frontiers in Dynamic Access Networks (DySPAN)*, 2008.

[31] M. McHenry, E. Livsics, T. Nguyen, and N. Majumdar, "XG dynamic spectrum access field test results [topics in radio communications]," *IEEE Communications Magazine*, vol. 45, no. 6, pp. 51–57, June 2007.

[32] R. Thomas, R. Komali, A. MacKenzie, and L. DaSilva, "Joint power and channel minimization in topology control: A cognitive network approach," in *IEEE International Conference on Communications*, June 2007, pp. 6538–6543.

[33] M. ElNainay, "Island genetic algorithm-based cognitive networks," Ph.D. dissertation, Virginia Polytechnic Institute and State University, 2009.

[34] V. Srivastava and M. Motani, *"Cross-layer design and optimization in wireless networks"*. Wiley, 2007, ch. Cognitive Networks: Towards Self-Aware Networks, pp. 121–146.

[35] S. Farrell, V. Cahill, D. Geraghty, I. Humphreys, and P. McDonald, "When TCP breaks: Delay- and disruption- tolerant networking," *IEEE Internet Computing*, vol. 10, no. 4, pp. 72–78, 2006.

[36] [Online]. Available: http://www.bbn.com/technology/networking/wnan

[37] L. DaSilva, A. MacKenzie, C. da Silva, and R. Thomas, "Requirements of an open platform for cognitive networks experiments," in *3rd IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks*, Oct. 2008, pp. 1–8.

[38] M. van der Schaar and F. Fu, "Spectrum access games and strategic learning in cognitive radio networks for delay-critical applications," *Proceedings of the IEEE*, vol. 97, no. 4, pp. 720–740, April 2009.

[39] J. Suris, L. DaSilva, Z. Han, and A. MacKenzie, "Cooperative game theory for distributed spectrum sharing," in *IEEE International Conference on Communications*, June 2007, pp. 5282–5287.

[40] D. H. Friend, R. W. Thomas, A. B. MacKenzie, and L. A. DaSilva, *"Distributed learning and reasoning in cognitive networks: Methods and design decisions"*. Wiley-Interscience, 2007, ch. Cognitive Networks: Towards Self-Aware Networks, pp. 223 – 246.

[41] M. Newman, A.-L. Barabasi, and D. J. Watts, Eds., *The Structure and Dynamics of Networks*. Princeton University Press; 1 edition, 2006.

[42] S. M. Mishra, A. Sahai, and R. W. Brodersen, "Cooperative sensing among cognitive radios," in *IEEE International Conference on Communications*, 2006, pp. 1658–1663.

[43] R. Jurdak, *Wireless Ad Hoc and Sensor Networks: A Cross-layer Design Perspective*. Springer, 1st Edition, 2007.

[44] F. Ge and C. W. Bostian, "A parallel computing based spectrum sensing approach for signal detection under conditions of low snr and rayleigh multipath fading," in *The 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2008.

[45] Y. Zhao, L. Morales, J. Gaeddert, K. Bae, J. Um, and J. Reed, "Applying radio environment maps to cognitive wireless regional area networks," in *In Proc. of the IEEE New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2007.

[46] E. Blossom, "Exploring GNU radio," November 2004. [Online]. Available: http://www.gnu.org/software/gnuradio/doc/exploring-gnuradio.html

[47] [Online]. Available: http://ossie.wireless.vt.edu/.

[48] J. Hennessy and D. Patterson, *Computer Architecture A Quantitative Approach*. Morgan Kaufmann, Fourth edition, 2006.

[49] A. Silberschatz, P. B. Galvin, G. Gagne, and A. Silberschatz, *Operating System Concepts*. Wiley; 6th edition, 2001.

[50] M. Ettus, "The universal software radio peripheral (USRP)," 2008. [Online]. Available: http://www.ettus.com,EttusResearch,LLC

[51] T. J. Brisebois, "Wideband RF front end daughterboard based on the motorola RFIC," Master's thesis, Virginia Polytechnic Institute and State University, 2009.

[52] P. F. Marshall, "Cognitive radio as a mechanism to manage front-end linearity and dynamic range," *IEEE Communications Magazine*, vol. 47, pp. 81–87, 2009.

[53] D. DePardo, J. B. Evans, J. A. Roberts, V. R. Petty, A. M. Wyglinski, P. J. Kolodzy, and M. J. Marcus, "Observations of potential secondary user device effects on digital television receivers," Technical Report ITTC-FY2008-TR-41420-06, 2007.

[54] M. Y. ElNainay, F. Ge, Y. Wang, A. E. Hilal, Y. Shi, A. B. MacKenzie, and C. W. Bostian, "Channel allocation for dynamic spectrum access cognitive networks using localized island genetic algorithm," *International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities*, vol. 0, pp. 1–3, 2009.

[55] T. W. Rondeau, "Application of artificial intelligence to wireless communications," PhD Dissertation Virginia Polytechnic Institute and State University, 2007.

[56] J. W. S. Liu, *Real-Time Systems*. Prentice Hall, 2000.

[57] R. J. Fong, S. J. Harper, and P. M. Athanas, "A versatile framework for fpga field updates: An application of partial self-reconfiguation," in *RSP '03: Proceedings of the 14th IEEE International Workshop on Rapid System Prototyping (RSP'03)*. Washington, DC, USA: IEEE Computer Society, 2003, p. 117.

[58] [Online]. Available: http://www.apcointl.org/frequency/project25/information.html.

[59] [Online]. Available: http://pagesperso-orange.fr/sebastien.godard/.

[60] [Online]. Available: http://www.vislab.uq.edu.au/users/manuals/python.2.4/lib/profile-instant.html

[61] R. A. Berry and E. Yeh, "Cross-layer wireless resource allocation," *IEEE Signal Processing Magazine*, vol. 21, pp. 59–68, 2004.

[62] "ADROIT: GNU Radio architectural changes," May 2007. [Online]. Available: http://acert.ir.bbn.com/downloads/adroit/gnuradioarchitectural-enhancements-3.pdf

[63] [Online]. Available: http://www.redpinesignals.com/.

[64] K. Medepalli, P. Gopalakrishnan, D. Famolari, and T. Kodama, "Voice capacity of ieee 802.11b, 802.11a and 802.11g wireless lans," in *IEEE Global Telecommunications Conference*, 2004.

[65] "Birth of broadband," International Telecommunications Union, September 2003.

[66] J. Neel, S. Srikanteswara, J. H. Reed, and P. M. Athanas, "A comparative study of the suitability of a custom computing machine and a vliw dsp for use in 3g applications," in *IEEE Workshop on Signal Processing Systems (SIPS)*, 2004.

[67] A. Nilsson, E. Tell, and D. Liu, "An accelerator structure for programmablemulti-standard baseband processors," in *Wireless Networks and Emerging Technologies*, 2004.

[68] T. Takano, H. Gambe, and T. Katoh, "Fujitsu's challenges in wireless communications," *Fujitsu Sci. Tech. J.*, vol. 38, pp. 121–133, 2002.

[69] H. Jiao, A. Nilsson, and D. Liu, "Mips cost estimation for ofdm-vblast systems," in *IEEE Wireless Communications and Networking Conference*, 2006.

[70] P. Athanas, J. Bowen, T. Dunham, C. Patterson, J. Rice, M. Shelburne, J. Suris, M. Bucciero, and J. Graf, "Wires on demand: Run-time communication synthesis for reconfigurable computing," in *International Conference on Field Programmable Logic and Applications (FPL)*, Amsterdam, Netherlands, 2007.

[71] J. A. Kahle, M. N. Day, H. P. Hofstee, C. R. Johns, T. R. Maeurer, and D. Shippy, "Introduction to the cell multiprocessor," *IBM Journal of Research and Development*, vol. 49, pp. 589–604, 2005.

[72] [Online]. Available: http://www.nvidia.com

[73] [Online]. Available: http://www.intel.com

[74] T. Schmid, O. Sekkat, and M. B. Srivastava, "An experimental study of network performance impact of increased latency in software defined radios," WiNTECH: The Second ACM International Workshop on Wireless Network Testbeds, Experimental evaluation and CHaracterization, Montreal, QC, Canada, 2007.

[75] F. F. Kuo, "The ALOHA system," *ACM SIGCOMM Computer Communication Review*, vol. 25, pp. 41–44, 1995.

[76] D. Willkomm, S. Machiraju, J. Bolot, and A. Wolisz, "Primary users in cellular networks: A large-scale measurement study," in *In Proc. of the 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2008.

[77] I. F. Akyildiz, W. Y. Lee, M. C. Vuran, and S. Mohanty, "NeXt generation/dynamic spectrum access/cognitive radio wireless networks: a survey," *Computer Networks: The International Journal of Computer and Telecommunications Networking*, vol. 50, pp. 2127–2159, 2006.

[78] S. D. Meinrath and M. Calabrese, "Unlicensed broadband device technologies: 'white space device' operations on the tv band and the myth of harmful interference," New America Foundation: Policy Backgrounder, 2007.

[79] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Wiley-Interscience, Second edition, 2000.

[80] R. Tandra and A. Sahai, "Snr walls for feature detectors," in *1nd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2005, pp. 559–570.

[81] O. Dobre, A. Abdi, Y. Bar-Ness, and W. Su, "Blind modulation classification: a concept whose time has come," in *IEEE/Sarnoff Symposium on Advances in Wired and Wireless Communication*, 2005.

[82] Z. Chen, Z. Hu, and R. Qiu, "Quickest spectrum detection using hidden markov model for cognitive radio," in *IEEE Military Communications*, 2009.

[83] B. Le, "Building a cognitive radio: From architecture definition to prototype implementation," Ph.D. dissertation, Virginia Polytechnic Institute and State University, 2007.

[84] E. Avci and D. Avci, "A novel approach for digital radio signal classification: Wavelet packet energy-multiclass support vector machine (wpe-msvm)," *Expert Systems with Applications: An International Journal*, vol. 34, pp. 2140–2147, 2008.

[85] Y. Wang, Q. Chen, and C. W. Bostian, "Universal classifier and synchronizer," *International Journal of Autonomous and Adaptive Communications Systems*, 2009, in press.

[86] W. A. Gardner and C. M. Spooner, "Signal interception: performance advantages of cyclic-feature detectors," *IEEE Trans. Commun.*, vol. 40, no. 1, pp. 149–159, 1992.

[87] M. Gschwind, "The cell broadband engine: Exploiting multiple levels of parallelism in a chip multiprocessor," *International Journal of Parallel Programming*, vol. 35, pp. 233–262, 2007.

[88] R. S. Roberts and J. H. H. Loomis, "Parallel computation structures for a class of cyclic spectral analysis algorithm," *Journal of VLSI Signal Processing*, vol. 10, pp. 25–40, 1995.

[89] A. C. Chow, G. C. Fossum, and D. A. Brokenshire, "A programming example: Large fft on the cell broadband engine," Global Signal Processing Expo (GSPx), Santa Clara, California, 2005.

[90] W. A. Gardner, A. Napolitano, and L. Paura, "Cyclostationarity: Half a century of research," *Journal of Signal Processing*, vol. 86, no. 4, pp. 639–697, 2006.

[91] R. S. Roberts, W. A. Brown, and J. H. H. Loomis, "Computationally efficient algorithms for cyclic spectral analysis," *IEEE Signal Processing Magazine*, vol. 8, pp. 38–49, 1991.

[92] W. A. Gardner, *Statistical spectral analysis: a nonprobabilistic theory*. Prentice-Hall, 1988.

[93] N. J. Carter, "Implementation of cyclic spectral analysis methods," Master's thesis, Naval Postgraduate School, 1993.

[94] F. Ge and C. W. Bostian, "A wide band spectrum sensing approach with agility and low snr sensitivity," Software Defined Radio Technical Conference, Denver, Colorado, 2007.

[95] E. L. d. Costa, "Detection and identification of cyclostationary signals," Master's thesis, Naval Postgraduate School, 1996.

[96] O. A. Yeste-Ojeda and J. Grajal, "Limitations of spectral correlation based detectors," in *IEEE/SP 14th Workshop Statistical Signal Processing*, 2007, pp. 244–248.

[97] D. H. Bailey, "A high-performance fast fourier transform algorithm for the cray-2," *The Journal of Supercomputing*, vol. 1, pp. 43–60, 1987.

[98] K. Kim, I. Akbar, K. Bae, J. Urn, C. Spooner, and J. Reed, "Cyclostationary approaches to signal detection and classification in cognitive radio," in *2nd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2007, pp. 212–215.

[99] W. C. Headley, J. Reed, and C. R. C. M. da Silva, "Distributed cyclic spectrum feature-based modulation classification," in *Proc. IEEE Wireless Comm. and Netw. Conf.*, 2008, pp. 1200–1204.

[100] P. Sutton, K. Nolan, and L. Doyle, "Cyclostationary signatures for rendezvous in ofdm-based dynamic spectrum access networks," in *2nd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2007, pp. 220–231.

[101] P. Sutton, J. Lotze, K. Nolan, and L. Doyle, "Cyclostationary signature detection in multipath rayleigh fading environments," in *2nd International Conference on Cognitive Radio Oriented Wireless Networks and Communications (CROWNCOM)*, 2007, pp. 220–231.

[102] M. C. Jeruchim, P. Balabanand, and K. S. Shanmugan, *Simulation of Communication Systems, Second Edition*. New York, Kluwer Academic/Plenum, 2000.

[103] W. A. Gardner, "Signal interception: A unifying theoretical framework for feature detection," *IEEE Transactions on Communications*, vol. 36, no. 8, pp. 897–906, 1988.

[104] E. Visotsky, S. Kuffner, and R. Peterson, "On collaborative detection of tv transmissions in support of dynamic spectrum sharing," in *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2005.

[105] A. Ghasemi and E. S. Sousa, "Collaborative spectrum sensing for opportunistic access in fading environments," in *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2005.

[106] R. R. Tenney and N. R. Sandell, "Detection with distributed sensors," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-17, pp. 501–510, 1981.

[107] Z. Chair and P. K. Varshney, "Optimal data fusion in multiple sensor detection systems," *IEEE Transactions on Aerospace Electronic Systems*, vol. 22, pp. 98–101, Jan. 1986.

[108] J. N. Tsitsiklis, "Decentralized detection," in *Advances in Statistical Signal Processing*. JAI Press, 1993, pp. 297–344.

[109] S. Aldosari and J. Moura, "Fusion in sensor networks with communication constraints," in *Third International Symposium on Information Processing in Sensor Networks*, April 2004, pp. 108–115.

[110] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *Automatic Control, IEEE Transactions on*, vol. 49, no. 9, pp. 1520–1533, 2004.

[111] A. Jadbabaie, J. Lin, and A. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *Automatic Control, IEEE Transactions on*, vol. 48, no. 6, pp. 988–1001, June 2003.

[112] F. Zhao, J. Shin, and J. Reich, "Information-driven dynamic sensor collaboration for tracking applications," *IEEE Signal Processing Magazine*, vol. v19, pp. 61–72, 2002.

[113] K. Ogata, *Modern Control Engineering*. Prentice Hall; 4 edition, 2001.

[114] Z.-Q. Luo and J. Tsitsiklis, "Data fusion with minimal communication," *IEEE Transactions on Information Theory*, vol. 40, no. 5, pp. 1551–1563, Sep 1994.

[115] A. Wun, M. Petrovi, and H.-A. Jacobsen, "A system for semantic data fusion in sensor networks," in *Proceedings of the 2007 inaugural international conference on Distributed event-based systems*, 2007, pp. 75–79.

[116] [Online]. Available: http://www.darpa.mil/sto

[117] "IEEE p802.22tm/d0.1 draft standard for wireless regional area networks," IEEE 802.22 doc. no. 22-06-0068-00-0000, Tech. Rep., 2006.

[118] S. Gandhi, C. Buragohain, L. Cao, H. Zheng, and S. Suri, "A general framework for wireless spectrum auctions," in *In Proc. of the IEEE New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2007.

[119] C. Cordeiro and K. Challapali, "C-mac: A cognitive mac protocol for multi-channel wireless," in *Proc. IEEE Dynamic Spectrum Access Networks Conference*, 2007.

[120] J. Hightower, R. Want, and G. Borriello, "Spoton: An indoor 3d location sensing technology based on rf signal strength," UW-CSE 00-02-02, University of Washington, Department of Computer Science and Engineering, Seattle, WA, February 2000.

[121] D. P. Agrawal, R. Biswas, N. Jain, A. Mukherjee, S. Sekhar, and A. Gupta, *Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless, and Peer-to-peer Networks*. CRC Press, 2006, ch. Sensor Systems: State of the Art and Future Challenges.

[122] J. Hennessy and D. Patterson, *Computer Architecture - A Quantitative Approach*, D. Penrose, Ed. Morgan Kaufmann, 2003.

[123] [Online]. Available: http://www.mysql.com/

[124] S. Anand, Z. Jin, and K. P. Subbalakshmi, "Belief propagation on factor graphs for cooperative spectrum sensing in cognitive radio," in *In Proc. of the 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2008.

[125] A. De Vany, "Implementing a market-based spectrum policy," *Journal of Law and Economics*, vol. 41, pp. 167–171, 1998.

[126] X. Zhou, S. Gandhi, S. Suri, and H. Zheng, "ebay in the sky: strategy-proof wireless spectrum auctions," in *Proceedings of the 14th ACM international conference on Mobile computing and networking*, 2008.

[127] Q. Chen, Y. Wang, and C. W. Bostian, "Universal classifier synchronizer demodulator," The 1st IEEE International Workshop on Dynamic Spectrum Access and Cognitive Radio Networks, December 2008, austin, Texas, USA.

[128] F. Ge, R. Rangnekar, A. Radhakrishnan, S. Nair, Q. Chen, A. Faye, Y. Wang, and C. W. Bostian, "A cooperative sensing based spectrum broker for dynamic spectrum access," in *IEEE Military Communications Conference (MILCOM)*, 2009.

[129] F. Ge, A. Radhakrishnan, M. Y. ElNainay, Q. Chen, C. W. Bostian, and A. B. MacKenzie, "Software radio-based decentralized dynamic spectrum access networks: A prototype design and experimental results," in *The 4rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2010, submitted.

[130] M. T. Ozsu and P. Valduriez, *Principles of Distributed Databases*. Pearson Education; 2 edition, 1999.

[131] P. Gupta and P. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, Mar 2000.

[132] M. Vu, N. Devroye, M. Sharif, and V. Tarokh, "Scaling laws of cognitive networks," in *2nd International Conference on Cognitive Radio Oriented Wireless Networks and Communications*, Aug. 2007, pp. 2–8.

[133] M. Y. ElNainay, D. H. Friend, and A. B. MacKenzie, "Channel allocation and power control for dynamic spectrum cognitive networks using a localized island genetic

algorithm," in *IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks*, Oct. 2008, pp. 1–5.

[134] F. Ge and C. W. Bostian, "SDR implementation issues: RF front end nonlinearity and dynamic computing resource allocation," in *Software Defined Radio Technical Conference*, Washington DC, 2009.

[135] T. Clausen, P. J. (editors), C. Adjih, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, and L.Viennot, "Optimized link state routing protocol (olsr)," RFC 3626, Network Working Group, October 2003.

[136] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint relaying for flooding broadcast messages in mobile wireless networks," in *Proceedings of the 35th Annual Hawaii International Conference on System Sciences*, Jan. 2002, pp. 3866–3875.

[137] [Online]. Available: http://openmaniak.com/iperf.php

[138] R. Frigg and S. Hartmann, "Models in science," in *The Stanford Encyclopedia of Philosophy*, summer 2009 ed., E. N. Zalta, Ed., 2009. [Online]. Available: http://plato.stanford.edu/archives/sum2009/entries/models-science/

[139] J. M. Ottino, "Engineering complex systems," *Nature*, vol. 427, p. 399, 2004.

[140] P. W. Anderson, "More is different," *Science*, vol. 177, pp. 393–396, 1972.

[141] R. B. Laughlin, D. Pines, J. Schmalian, B. P. Stojkovic, and P. Wolynes, "The middle way," *PNAS*, vol. 97, pp. 32–37, 2000.

[142] "Grand challenges for engineering," National Academy of Engineering, Tech. Rep., 2008.

[143] R. Ananthanarayanan, S. K. Esser, H. D. Simon, and D. S. Modha, "The cat is out of the bag: Cortical simulations with 10^9 neurons, 10^{13} synapse," in *International Conference for High Performance Computing, Networking, Storage and Analysis*, 2009.

[144] B. Horine and D. Turgut, "Link rendezvous protocol for cognitive radio networks," in *New Frontiers in Dynamic Spectrum Access Networks, 2007. DySPAN 2007. 2nd IEEE International Symposium on*, April 2007, pp. 444–447.

[145] L. DaSilva and I. Guerreiro, "Sequence-based rendezvous for dynamic spectrum access," in *New Frontiers in Dynamic Spectrum Access Networks, 2008. DySPAN 2008. 3rd IEEE Symposium on*, Oct. 2008, pp. 1–7.