Metronome: Coordinating Spectrum Sharing in Heterogeneous Wireless Networks

Ramakrishna Gummadi* Hari Balakrishnan* Srinivasan Seshan†
*MIT CSAIL † CMU

***Abstract*—Many licensed and unlicensed frequency bands support heterogeneous wireless networks running different physical and link layer protocols. These networks "share" spectrum, but in an anarchic and arbitrary manner, resulting in poor performance for some networks and sub-optimal performance in aggregate. This problem is likely to be of importance in the US 700 MHz TV band, which is being explored for secondary use.**

This paper describes *Metronome*, a system that allows heterogeneous networks to coexist well. Metronome provides a flexible and expressive policy language that allows a network operator to specify constraints on receiver performance metrics such as throughput or loss rates. Metronome then configures each participating transmitter with appropriate channel, bandwidth, and transmission power settings automatically. Experiments from an outdoor vehicular platform for monitoring the TV band, and from an indoor heterogeneous network of 802.11, ZigBee and Bluetooth devices demonstrate the utility of Metronome's policy language. In a network of coexisting devices, we find that Metronome improves the throughputs of ZigBee and Bluetooth by more than 6× and that of 802.11 by more than 15%.

I. Introduction

An increasingly popular way to cope with spectrum scarcity is to allow heterogeneous networks to share frequency bands. Unlicensed bands (such as the 915 MHz and 2.4 GHz bands) have always had many different networks operate concurrently, but this shared use has led to performance problems [13, 27] that could worsen in the future. In addition, heterogeneous wireless networks are now being actively considered for the licensed bands [14, 15] because these bands are often vastly under-utilized by the assigned (primary) network, even in densely populated areas [21]. For example, the US Federal Communications Commission (FCC) has recently issued a notice [11] to open up licensed spectrum in the 700 MHz TV bands to networks that can use this spectrum on a secondary (i.e., non-interfering) basis. The FCC has also recently opened up the 3.65–3.7 GHz band under "lite" licensing conditions that allow high-powered 802.11 devices to operate as secondary transmitters.

This paper addresses a key question inspired by these trends: how should heterogeneous wireless networks operating in a band coordinate with each other to share spectrum? Within a single network, this task is done using a media access (MAC) protocol and packet scheduling, but this approach does not extend to situations where there is no common physical or link layer, and where a network provider might not be able to control all of the devices (e.g., television transmitters and receivers, or competing network providers' nodes). In addition, the right answer to this question depends on the context. For example, an ISP operating in the 700 MHz band might be interested foremost in reducing interference from other ISPs using the same band. In contrast, for other forms of sharing, one might be interested in ensuring high aggregate throughput, or in prioritizing some devices over others (e.g., all Bluetooth headsets should get a certain bit-rate), or in ensuring that each type of network maximizes its throughput while not causing more than a certain level of interference to other networks.

The first contribution of this paper is to develop a simple policy language that allows an administrative entity to express a rich set of policies specifying the throughput, loss-rate and interference requirements on the networks it manages. For example, an ISP operating in the 700 MHz band might want to use channels and power levels that do not interfere with any TV receivers or other ISPs' transmitters, while an operator of a ZigBee-based sensor network might want to ensure that none of its 802.11 transmitters interfere with the ZigBee nodes. There is a need for such a policy language because bodies such as the IEEE 802.22 working group are only standardizing mechanisms for detecting primary transmitters such as TV signals and microphones within the 700 MHz band. So, competing ISPs that wish to use this band must still specify their own policies so as to reduce mutual interference among themselves.

Our second contribution is to identify and implement a small core set of mechanisms that allows several different policies to be expressed flexibly. These mechanisms include an extensible technique to detect activity in any band of the relevant spectrum, the basic operations that each participating transmitter should support, and a control protocol to periodically communicate information about spectrum activity and to inform each participating transmitter about its operating channel, bandwidth, and power.

We have implemented and evaluated our policy language and attendant mechanisms as part of the *Metronome* system. Our policy language takes the form of SINR requirements that must be maintained in order to maintain the desired link throughput and loss rates. Most current wireless devices operate only in narrow-band channels and cannot sense the spectrum beyond simple energy detection within their operational band. However, because the usable spectrum is typically much larger, we deploy wideband-capable spectrum sensors in the form of software radio monitors to gather wide-band spectrum activity information and assist in enforcing SINR policies. We exploit monitor mobility to reduce the cost of monitor deployment, increase the size of the monitored area, and reduce the hidden terminal problem in which a monitor sees a very different signal quality than an actual receiver due

to RF propagation effects.

We have implemented Metronome using the Universal Software Radio Peripheral (USRP) hardware and GNURadio software for the monitors. We evaluate Metronome on a heterogeneous network in the 2.4 GHz band consisting of 802.11b/g, Bluetooth and ZigBee scattered on one of the floors of our office building.

We find that Metronome improves the throughputs of ZigBee and Bluetooth by more than 6× and that of 802.11 by more than 15% using only simple policies such as reserving minimum SINRs for ZigBee and Bluetooth, while letting 802.11 optimize the remaining capacity.

II. Problems due to Interference

Metronome implements three key concepts to detect signals and combat interference across heterogeneous networks: (i) a flexible policy framework for computing good transmitter settings; (ii) a matched filter-based detection for separating out signal power of one particular network from interference power of all other networks; and (iii) mobile monitors for collecting multiple spatially distributed samples of signal and interference levels around a receiver in order to robustly compute the signal and interference powers.

We first describe the testbeds used for measuring signals and interference and then show why we need to go beyond simple energy detection to identify and reduce interference.

A. Background and Experimental Setup

For our indoor testbed, we deployed 802.11, ZigBee and Bluetooth devices, and monitored the interference generated by one type of device at another device type, as well as the interference from external transmitters that are not under Metronome's control. There are two transmitters and two corresponding receivers for each device category (i.e., 802.11, ZigBee and Bluetooth), and these devices are placed randomly and moved around on a large office floor. We deployed the indoor testbed over a period of one month, and observed more than 200 unique devices of various types.

For 802.11, we used Atheros AR5212 cards operating in the 2.4 GHz band. There are 11 overlapping 802.11 channels in this band. The center frequency F_c of each channel is given by $F_c = 2412 + 5c$ MHz, where $c \in [0,10]$ is the channel number. The bandwidth of 802.11 is 22 MHz for 802.11b and 20 MHz for 802.11g. For ZigBee, we used Chipcon CC2420 transceivers, which are mounted on CC2420DBK development kits in order to accurately transmit and receive packets with low overhead, and measure link loss. ZigBee operates in one of 15 non-overlapping channels in the 2.4 GHz band. The center frequency F_c of each channel is given by $F_c = 2405 + 5c$ MHz, $c \in [0,14]$, so 802.11 and ZigBee can fully overlap. The bandwidth of each ZigBee channel is 3 MHz. Finally, the Bluetooth transmitters and receivers we used were Belkin USB Bluetooth controllers and headsets. Bluetooth uses FHSS (Frequency Hopping Spread Spectrum) across 79 channels, each 1 MHz wide, also in the 2.4 GHz band. The hop rate is 1600 times per second, which is designed to provide immunity to localized interference. Both ZigBee and Bluetooth transmit at a much lower power (1 mW) than 802.11 (80 mW).

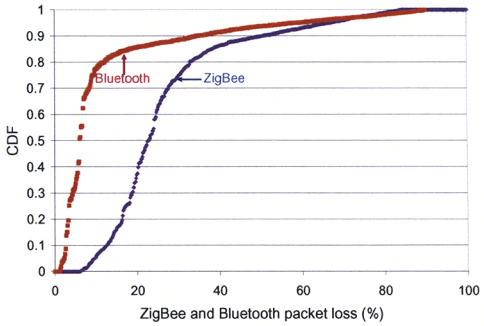

Fig. 1. Effect of 802.11 interference on Bluetooth and ZigBee loss rates.

The workload consists of TCP transfers for 802.11, latency-sensitive continuous voice traffic for Bluetooth, and loss-sensitive continuous sensor traffic for ZigBee. The metrics are average completion latency for 802.11, and average link loss rates for Bluetooth and ZigBee. We don't modify the default link parameters for ZigBee, and measure the link loss rates reported by the kernel.

Despite using some form of spread spectrum, there is some evidence of significant interference-related problems with both ZigBee and Bluetooth. The effect of 802.11 interference on Bluetooth has been extensively quantified [13, 18, 27]. In fact, interference problems are severe enough that the Bluetooth standard now has a mode called Adaptive Frequency Hopping (AFH), in which it can first scan channels for interference and selectively use channels that are cleaner. Unfortunately, this scanning does not work if the Bluetooth transceiver is close to a strong interference source such as an 802.11 laptop, because the RF front-end is driven to saturation by 802.11. This phenomenon is called front-loading, and there are no satisfactory approaches to solve this problem.

B. Observations

The impact of interference on 802.11, ZigBee and Bluetooth is qualitatively different. First, because of the large difference in relative powers, when 802.11 interferes with ZigBee or Bluetooth, the link degradation is more severe on ZigBee and Bluetooth. Second, these networks respond differently to interference because of the nature of their workloads. The main effect of interference on 802.11 is to increase the completion latency of short TCP connections by several seconds. On the other hand, for ZigBee and Bluetooth, which carry loss-sensitive or latency-sensitive traffic, the primary impact is in the form of unacceptable signal or voice quality degradation. We observed link losses of more than 85% for ZigBee and Bluetooth under moderate to heavy interference (i.e., when there is severe SINR degradation, which happens when 802.11 and Bluetooth are close to each other).

Link degradation due to interference. We place the ZigBee and Bluetooth devices at various random locations spread around our office floor over an area of 10,000 sq. ft., while ensuring that packets can be exchanged with negligible loss

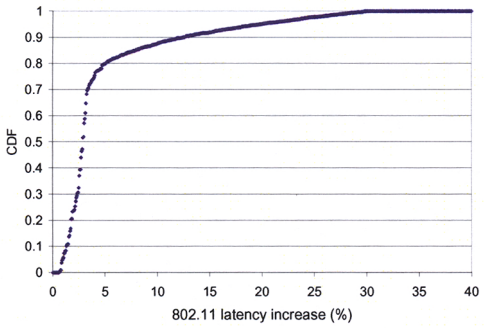

Fig. 2. Effect of ZigBee and Bluetooth interference on 802.11 latency.

between each transmitter/receiver pair. We then move the 802.11 devices around so that they can interfere with ZigBee and Bluetooth. We thus obtained more than 1000 readings for the link performance of each protocol under interference. We measure the link losses on ZigBee and Bluetooth due to interference from 802.11 (Figure 1), as well as the increase in TCP latency of 802.11 due to interference from Bluetooth and ZigBee (Figure 2). Figure 2 shows that the median ZigBee loss rate exceeds 20%, which significantly degrades the quality of collected sensor samples because existing ZigBee implementations do not implement end-to-end reliability due to severe resource constraints (such as typically having only 4 KB RAM). Also, Bluetooth can exceed a loss rate of 20% in more than 10% of cases, which renders voice sessions on Bluetooth headsets incomprehensible. Similarly, from Figure 2, 802.11 sees a median latency increase of more than 5%.

Interference isolation through energy detection. Given the severe impact of interference on ZigBee and Bluetooth and a measurable impact on 802.11, the question that arises is how well it is possible to detect and isolate such interference. Today's approaches revolve mainly around using an 802.11 scanner to troubleshoot interference between 802.11 devices, or a spectrum analyzer to measure background noise and interference levels. Unfortunately, energy detection cannot identify the offending transmitter. For an example scenario involving interference between 802.11 and Bluetooth, Figure 3 shows the spectrum map captured using an Advantest R3265A spectrum analyzer that implements energy detection. There are two 802.11 transmitters on two neighboring channels (Channels 4 and 5), and a single Bluetooth receiver. The 802.11 transmitter on Channel 4 causes the most interference to ZigBee, but this fact cannot be deduced from the observed spectrum alone.

Manually changing transmitter settings. To eliminate interference, it is possible to manually change the channels used by 802.11 and ZigBee. For interference to be minimized, their spectra must not overlap. However, since there can be external 802.11 transmitters on adjacent channels, switching ZigBee or 802.11 channels so as to only ensure mutual spectrum separation is insufficient. Instead, in order to set the channels optimally, we should choose channels that maximize SINRs for both ZigBee and 802.11. Unfortunately, this is a tedious process because there are $11 \times 15 = 165$ possible 802.11 and

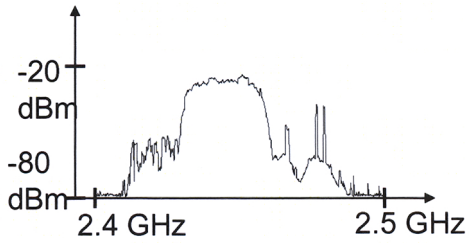

Fig. 3. Observed spectrum of Bluetooth and two 802.11 transmitters using energy detection.

ZigBee combinations. This process is not manually tractable as the number of devices grows.

III. Metronome Design

A Metronome user deploys monitors over any area where she is concerned that some wireless receiver's performance may degrade because of interference from other networks. The user also expresses a sharing policy in the policy server in the form of constraints on the SINR levels that different receivers (i.e., the co-located monitors) should see. The monitors continuously sample the energy across the band of interest, use a parameterized matched filter to associate that activity with a device type (e.g., 802.11, Bluetooth, etc.), and periodically send that information to the policy server. The policy server uses these monitor reports to calculate the interference contributions of each transmitter, as described in §III-B. The policy server then runs an optimization procedure using this individual transmitter interference information and the specified policy to determine the best transmit power and channel settings (center frequency and bandwidth) for the participating transmitters. The server sends these settings to the transmitters, which modify their behavior accordingly. This parameter setting ability allows Metronome to not only coordinate sharing between heterogeneous transmitters, but also optimize the transmitters under its control around fixed interference that is outside its control. Figure 4 illustrates this architecture. In this figure, Metronome ensures that its transmitters don't use any TV channels withing a region that are sensed as being used by external TV transmitters (TV transmitters are not shown).

Metronome separates the mechanisms for identifying which transmitters contribute to observed interference from the policy of what to do about it. It departs from traditional channel access wisdom by proposing a centralized policy engine to compute the right transmission parameters for each participating transmitter, rather than developing a fully distributed solution where each transmitter makes its own decisions. This centralized approach allows us to produce a better throughput allocation than distributed approaches by providing a more consistent and complete interference picture. Metronome also assumes that expressing constraints on SINR leads to useful policies; we experimentally validate this assumption with 802.11, ZigBee, and Bluetooth in §V.

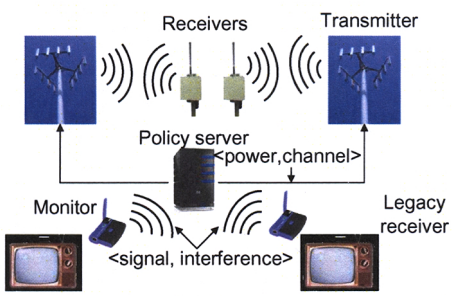

Fig. 4. Metronome architecture showing monitors, transmitters and the policy server. The monitors send SINR information to the policy server, which then sends messages to each transmitter setting its channel and power level. The process runs in the background, reporting information every few minutes and effecting changes to the transmitters at that time-scale.

A. Requirements and Assumptions

The main requirement, of course, is that Metronome allow heterogeneous wireless networks that share common spectrum (and nothing else) to co-exist well. By "co-exist well", we mean "allow different networks to each provide reasonable service when measured over time scales of minutes", rather than finer-grained throughput or fairness goals that are more typical for channel access protocols in a homogeneous network. We identify other main requirements below:

1) **Support different policies.** As explained in the introduction, different policies make sense in different contexts, even for any given heterogeneous mix. To constrain the design space, Metronome allows policies that depend on SINR, rather than other notions of performance such as throughput, latency, etc., because all these other performance metrics depend on SINR. We don't claim, however, that SINR captures all interesting policies, but rather that they capture useful ones (see §IV).
2) **Capture receiver-perceived interference.** Since we care about performance at receivers, it is important to capture interference conditions near them. Radio propagation is complicated and not easy to model, so rather than rely on some model of propagation, we use mobile monitors that collect signal and interference samples at several spatially distributed locations around the receiver. We then use the average signal value computed across all such samples as the receiver's signal level, and the maximum interference value across samples as the receiver's interference level. Averaging signal values reduces the impact of small-scale fading effects such as multipath that are different at the receiver and the monitor, while using the maximum interference value reduces the impact of a missed interference event because the monitor temporarily became a hidden terminal with respect to the interferer due to shadowing or fading, and is thus unable to observe the interferer's impact at the receiver. In §V, we show that this approach does, in fact, allow mobile monitors to accurately capture the signal and interference levels experienced at a receiver.
3) **Support legacy communication systems.** It is virtually impossible to go about modifying many existing transmitters and receivers, such as televisions. In licensed bands that open up to secondary users, we assume that the primary network is unmodified, but that we can modify the behavior of secondary transmitters dynamically, changing their transmit power and channel. The main goal is to ensure that secondary users can operate while ensuring that the SINR of the primary receivers is controlled.
4) **Support evolution.** Recently, we have witnessed a spate of new wireless protocols in both licensed and unlicensed bands, such as IEEE standards 802.11y-2007 (High-power 802.11), 802.16 (WiMAX), 802.20, 802.22, 802.15.3 (Ultra-Wideband), and 802.15.4 (ZigBee), apart from a slew of non-standardized protocols. The proposed architecture must accommodate new protocols as they become viable without requiring modifications to other participating (and legacy) networks.
5) **Control individual participating transmitters.** Our goal is to control the transmission parameters of each participating transmitter. To achieve this goal, Metronome needs to measure interference and map it to individual transmitters as accurately as possible. To scale well and be manageable, this process should not assume that the locations of the transmitters or receivers are known.

Metronome makes a few assumptions, some of which we have mentioned already, which we state here explicitly. First, a single administrative entity expresses the coordination policy. This restriction allows us to develop a simple architecture and to show that interesting, practical policies can be realized with our simple mechanisms. Second, participating transmitters can dynamically change their power levels, channel center frequency, and channel bandwidth within some operating range that is known to the policy server—for example, our Atheros 802.11g nodes can use more than 11 overlapping channels in the 2.4 GHz band, and the channel widths can be 5, 10, 20 or 40 MHz (even though 802.11g is required to only support 11 20 MHz-wide channels). In addition, any clients of a transmitter can also adapt to these configuration changes. Many networks such as 802.11 and Wireless USB can deal with such a reconfiguration event, because their clients provide automatic scanning and association functionality that is also used during mobility. Networks using frequency hopping, such as the DECT standard for cordless phones and baby monitors, also provide built-in scanning support. Third, the participating transmitters have IP network connectivity to the policy server, so that it can adjust transmit parameters, by leveraging the fact that many wireless devices ultimately have a wired path to the Internet, either directly as in the case of APs, or indirectly, as in the case of Bluetooth devices connected to laptops, etc. However, we allow monitors to collect timestamped samples offline and upload them to the policy server, as in the case of 700 MHz monitors. Fourth,

the user is able to specify for each receiver a set of SINR constraints to achieve. For legacy receivers, a threshold SINR above which it can operate properly is a good choice because it is usually available. Finally, Metronome operates at time scales of minutes, improving co-existence at time scales where persistent problems are likely to be observed.

B. Measuring Interference Contributions

Once a set of monitors is deployed and policy specified in the policy server, the system is ready to start running. Initially, each participating transmitter has some frequency channel and power level at which it runs. Every time a monitor detects a transmitter of a particular network type, it senses the energy, and calculates an average of this energy over some time interval, τ (set to 120 seconds in our experiments). Thus, for each network type, each monitor estimates the average signal level. The average interference level is the sum of average signal levels from all other network types detected by the monitor.

Under normal conditions, when the average interference levels for each network type from other heterogeneous networks observed by the monitor is below a specified threshold, the monitor does not send any information to the policy server. For example, in a primary-secondary setting, this threshold may be set to be no larger than the interference level tolerable to the primary. The monitor doesn't consider concurrent activity from devices of the same type to be interference, leaving that to the MAC layer of the individual networks.

In addition, the monitor and policy server together need to identify the individual transmitters responsible for causing significant interference. Metronome performs this task by first finding the set of potential interferers, which in general is a smaller set than the total number of transmitters in the system. This first step of finding potential interferers at a monitor therefore makes the second step of identifying individual transmitters and their interference contributions scalable and practical. This scalability is especially important in dynamic environments in which the deployed monitors observe a changing set of interfering transmitters over time.

Finding potential interferers. Periodically, the policy server requests the participating transmitter to broadcast a special beacon packet using the current setting of its channel parameters. The server ensures that no two participating transmitters in a given region send beacons concurrently. The monitors know the beacon schedule, so they know when to expect a beacon. Each monitor that can sense activity at the time that a beacon was sent reports the received signal strength to the policy server. Thus, the policy server knows which set of monitors can detect each transmitter. Note that we do not require the monitors to demodulate or correctly receive the beacon; rather, the monitor must simply be able to detect the presence of the broadcast beacon, which is a good indicator that the corresponding transmitter is a potential interferer. This beaconing scheme is robust in the face of concurrent activity seen at a monitor from transmitters of other network types, because the policy server knows the beaconing transmitter and its type. Concurrent activity from transmitters of the same type is not a problem because it is regulated by the native MAC.

Protocol	Modulation	n-ary	Differential	Quadrature	Pulse Shape
ZigBee	PSK	2	yes	half	Sine
Bluetooth	FSK	2	no	none	Gauss
1 Mbps 802.11b	PSK	2	yes	$\frac{\pi}{4}$	Rect
2 Mbps 802.11b	PSK	4	yes	$\frac{\pi}{4}$	Rect
802.11a/g	QAM	16/64	no	none	Rect

TABLE I
MODULATION PARAMETERS FOR DIFFERENT PROTOCOLS.

In order to detect beacons emitted by heterogeneous transmitters and measure their signal strength, the monitors use a parameterized matched filtering technique. The key goal behind parameterized matched filtering is to look for waveforms known *a priori*, such as preambles in the case of data packets, or pilots in the case of analog signals such as TVs. With matched filtering, the incoming samples are processed and convolved against a locally generated waveform of the known preamble. If the output of the convolution is high, the monitor knows that a transmission with a certain preamble type is in progress. The monitor measures the strength of the samples at the output of the correlator, from which it estimates the strength of the desired signal in the samples. Thiss ignal strength estimation is both sensitive (i.e., low-power signals can be detected) and accurate because the correlation operation suppresses interference from other networks. By continuously sampling the medium and correlating the samples against waveforms from the various networks being monitored, a monitor generates an accurate map of signal levels of the networks active in the region. Furthermore, the monitors can compute the interference experienced by receivers of a given network as the sum of all signal levels of all other networks.

The main challenges in implementing such a mechanism are that the filtering technique must be flexible (i.e., be able to easily specify multiple wireless protocols), extensible (i.e., be able to add new protocols in future), and must cope with processing and memory limitations that arise in simultaneously looking for waveforms from multiple network types.

The chief observation we make to provide monitor flexibility and extensibility is that most protocols pick from a small palette of well-known modulation techniques that are parameterized in various ways. We identify the primary parameters used for modulating a carrier: the class, order, and type of modulation and the nature of pulse shaping and filtering applied to the carrier wave. Our matched filter is parameterized by these values for several common network protocols, as shown in Table I.

A final challenge in measuring interference is that the monitors must capture the interference levels experienced at the receivers. However, because of RF propagation characteristics, the interference measurements at a monitor can differ significantly from the interference at the receiver. In extreme cases, the monitor might not even be able to measure any interference experienced at the receiver, resulting in the classic "hidden terminal" problem.

We use monitor mobility as a way to mitigate this problem. Mobile monitors overcome the hidden terminal problem by computing the average signal level and the maximum interference level across several spatial samples. Mobile monitors also have the advantage that they are physically separate from receivers, which means legacy primary receivers like TVs and wireless microphones need not be modified in order to measure the interference they experience from secondary TV band transmitters.

Identifying individual transmitters. When a monitor observes that the average interference has exceeded a desired threshold, it asks the policy server to instruct each of the monitor's interfering transmitters (computed as explained in the previous paragraphs) to start maintaining a timestamped record of the transmissions it makes (the time and duration). The monitor also timestamps all interference activity it detects. The interfering transmitters send the list of times at which they initiated transmissions, and the monitor sends the list of times at which detected activity, as well as the received signal strength of each detection. By correlating these two sets of data over a period of time, the policy server can associate any given detection with the transmitters responsible for it (if there is concurrent activity, multiple transmitters are responsible and will be detected in most cases). Because this scheme operates only when there is an anomalous condition at some monitor, the overhead at the monitor and transmitters arises only when it is necessary to obtain the information and change the behavior of some of the transmitters.

IV. Expressing and Enforcing Policy

In this section, we describe how the policy server allows various policies regulating the sharing of spectrum to be specified, and how it then computes suitable transmission parameters that include the transmission powers and channels of each of its transmitters.

A. Policy Expression

A Metronome policy is a set of rules and a single goal. The rules are linear inequalities on SINRs of monitors, and the goal is a linear function of the SINRs of one or more receivers. The $SINR[i,m,c]$ of a monitor m from a transmitter i on channel c is calculated as the signal level from the transmitter minus the interference levels from all other transmitters that are not of the same type as i. This is because transmitters of the same type are handled by the native MAC. The inequalities for the rules are upper and lower bounds on the SINR of any monitor m, and are of the form: $\text{SINR}_{\min}[i,m,c] \leq \text{SINR}[i,m,c] \leq \text{SINR}_{\max}[i,m,c]$. Either or both bounds can be left unspecified for a monitor. The goal of the policy is to maximize a desired linear function on the SINRs of some or all devices. The policy server then computes the transmitters' powers and channels to maximize the policy goal subject to the rules.

Since the performance of a receiver in the presence of interference depends on the SINR it observes, this approach for expressing policies is reasonable.

The main advantages of using SINRs as the basis for specifying policies is that the resulting policy framework is flexible, general, and efficiently solvable. It is flexible because different policies can be concisely expressed as linear rules and a goal on SINRs once requirements on maximizing throughput or minimizing loss rates are mapped into constraints on SINRs. This mapping is possible because, for any given modulation, it is possible to derive the nominal throughput and the error rate from standard SINR curves [24]. We don't impose any limitations on the form of the resulting rules and the goal, other than requiring them to be linear. This linearity requirement is justified because throughput per Hertz and SINR are monotonically related. In particular, assuming an AWGN (Additive White Gaussian Noise) channel, by Shannon's theorem, the maximum capacity $C = B\log(1 + SINR)$, where B is the channel bandwidth. Hence, maximizing an SINR-based policy goal also maximizes throughput.

The framework allows the SINR constraints to be specified at a fine granularity (at the level of individual receivers). and for any network type. It is efficiently solvable because the algorithm to implement policies uses linear programming.

We illustrate how policies can be flexibly specified for two different scenarios.

Coexisting with legacy TV transmitters and receivers at 700 MHz. The FCC spectrum policy for using the 700 MHz bands specifies that legacy devices such as TV receivers or wireless microphones should not be affected by secondary users. Potential secondary users include ISPs who want to provide wide-area high-bandwidth Internet access using this portion of the spectrum. There are several TV channels available in this band, and the FCC mandates that a channel can be used by secondary users only when no active TV transmission is detected on that channel. Both analog and digital TV transmissions use well-known pilots or preambles, so the goal for the ISP is to deploy mobile monitors within its service area so that a monitor can check for the pilot TV signal on each channel before the ISP can opportunistically use it. This scenario presents a compelling reason for monitors to use parameterized matched filtering over the narrow-band pilot signals because simple wide-band energy detection doesn't offer enough detection accuracy.

The ISP can then define the following policy, which respects the TV non-interference criterion imposed by the FCC. The ISP's transmitters must select a channel only when all monitors within a certain region report that any detected TV signal levels are *below* the reception threshold of a standard legacy TV receiver (which is specified to be -85 dBm [15]). We can translate this signal level into the SINR ratio through standard calculations for thermal noise power, which works out to be -104 dBm. We thus obtain the SINR constraints for this policy: the SINR of TV signals at every monitor should be below $-85-(-104) = 19$ dB.

Policy constraints : $\forall$ TV Transmitters $i, \text{SINR}[i,j,c] \leq 19$ dB.

In using a channel, an ISP wants to additionally minimize the interference from other secondary ISPs using the same

channel. Put another way, the ISP would like to maximize its own SINR subject to the TV-transmitter constraint. Current standards such as 802.22 don't provide an easy way to express this policy. So, the goal is to find the powers and channels of the ISP's transmitters so that the SINR of the ISP's own receivers is maximized, given the SINR constraints. Thus, we have to find the channel $c[j]$ from a given set of potential channels and the power $p[j]$ for each transmitter i so that

$$\textbf{Policy goal}: \max_{c[],p[]} \sum_{j} \text{SINR}[i,j,c].$$

Guaranteeing minimum throughput for ZigBee or Bluetooth. As shown in §II, we need to ensure that ZigBee or Bluetooth do not experience heavy losses in the presence of higher-powered transmitters such as 802.11. The simplest way to do this is to ensure that Bluetooth or ZigBee are always guaranteed a certain maximum loss rate, while asking 802.11 to maximize its throughput from the leftover spectrum. Thus, the policy's rules specify some minimum thresholds on the SINRs of Bluetooth and ZigBee receivers, and the policy's goal is to maximize the SINR of 802.11 receivers subject to the policy's constraints. For example, in order to maintain the packet error rate of Bluetooth voice below 10% (and thereby bound latency due to retries to the tolerable limit), we calculate that we need an SINR of 18 dB. So, the policy constraints are:

$$\textbf{Policy constraints}: \forall \text{ Bluetooth rcvrs } j, \text{SINR}[i,j,c] \geq 18 \text{ dB},$$

and the policy goal is to maximize SINR of all 802.11 receivers, which means finding a channel $c[i]$ and power level $p[i]$ for each 802.11 transmitter i so that

$$\textbf{Policy goal}: \max_{c[],p[]} \sum_{j} \text{SINR}[i,j,c].$$

Interestingly, as will be clear from the transmitter parameter selection algorithm in §IV-B and from the results in §V, the outcome of this policy on a network with 802.11b/g and ZigBee is different than on a network with 802.11b/g and Bluetooth. This is because Bluetooth already hops frequencies within the entire 2.4 GHz band, so the only possibility for 802.11b/g is to decrease its transmit power. On the other hand, because ZigBee occupies a fixed narrow-band channel, a heterogeneous network with 802.11b/g and ZigBee can also make 802.11b/g change channels without decreasing transmit power.

While we have described two representative policies in detail, we have implemented several other policies, such as enumerating all interfering transmitters, identifying external interference, and restricting the set of transmission parameters that can be varied (such as channels only, power only, center frequency only, etc.). We evaluate some of these policies in §V.

B. Enforcing Policy

As described in §III-B, the monitors calculate the signal and interference powers from each transmitter and report them to the policy server. Since the policy server knows the transmit

Algorithm IV.1: ENFORCEPOLICY($H[i][j][c]$)

```
for i ← 1 to N
do { for c ← c_min to c_max
     do { for j ← 1 to N
          do { if i = transmitter(j)
               then (i)signal[j] ← H[i][j][c]p[i]
               if i = heterogeneous(j)
               then (ii)interfer[j] += H[i][j][c]p[i]
               SINR[i,j,c] = signal[j] − interfer[j]
          Find      p[i]
          (iii) ∋   max policy(SINR[i,j,c])
          (iv)given ∀i p_min[i] ≤ p[i] ≤ p_max[i]
          (v)&      SINR_min[] ≤ SINR[] ≤ SINR_max[]
return (c[i],p[i] ∋ max_{c[],p[]} policy(SINR[]))
```

Fig. 5. LP-based algorithm for solving policies.

power of a transmitter, it can compute the channel attenuation factor $H[i][j][c]$ from transmitter i to receiver j for a given channel c the transmitter is on. The policy server uses these channel attenuation values as arguments to the policy solver algorithm shown in Algorithm 4.1 (i.e., Figure 5). The real work in Figure 5 is done by the LP solver (used in lines (iii)–(v)).

Algorithm 4.1 takes as input the $H[i][j][c]$ channel attenuation matrix, which represents attenuation from transmitter i, which is on channel c, to monitor j. We permit H to be a function of the transmitter frequency because real-life environments exhibit some amount of frequency-selective fading. If the monitor has not yet measured the interference power for some channel c a transmitter can be on because the transmitter was not asked by the policy server to switch to c, the policy server uses the average H value on channels that have been measured. Since the policy server re-runs Algorithm 4.1 when transmitter channels are adjusted and new measurements from the monitors become available, it is robust to missing $H[i][j][c]$ values for some channels c.

Algorithm 4.1 works as follows. The policy server iterates through all possible participating transmitters i in the outermost `for` loop, and considers each transmitter channel and all receivers in turn in the two inner loops. For each receiver j, it computes the received signal power as well as the sum of interference powers from transmitters of network types different from the receiver, in lines (i) and (ii). It excludes interference from devices of the same type because the native MAC handles this situation. Note that the resulting quantity, $\text{SINR}[i,j,c]$, is a linear combination of the unknown transmitter power variables $p[i]$.

For each channel, the server then solves for the transmitter power variables that maximize the policy goal specified in line (iii). The transmitter powers are required to be in the range $[p_{\min}[i], p_{\max}[i]]$, which capture the constraints on the output power of a device. We allow transmitter power to

vary at the granularity of an individual transmitter, because each transmitter could be from a different manufacturer. The policy rules are reflected in line (v). The policy server uses LP to solve for the transmit powers for each channel. It finally returns the channels and the corresponding transmission powers that maximize the policy goal across all channels.

The complexity of this algorithm is the complexity of LP times the number of transmitters and channels. The expected case performance of LP is linear [10], so our algorithm is efficient in practice.

V. Evaluation

We implemented Metronome using the GNURadio software radio platform for the monitors, and evaluated it on an indoor 2.4 GHz testbed as described in §II. We use the Universal Software Radio Peripheral (USRP) with appropriate RF front-end for the 2.4 GHz (RFX2400) bands. Due to bandwidth limitations of the USB2 bus used by the USRP of only up to 32 MB/s, we monitor a fixed 6 MHz-wide spectrum, which is sufficient to capture ZigBee and Bluetooth signals. While Bluetooth continuously hops frequencies within a 79 MHz-wide band, its bandwidth is only 1 MHz, which allows us to capture its signal fully with a duty-cycle of more than 7.5%, which is sufficient to estimate its signal and interference contributions. The spectrum is also wide enough to capture the PLCP preambles used by 802.11b/g, which are always sent at 1 Mbit/s regardless of the bit-rate of the actual packet. The parameterized matched filter is implemented using a combination of GNURadio components and MATLAB. We use code from [4, 5, 16, 28] as a starting point for our 802.11, ZigBee and Bluetooth implementations.

The policy server uses the LP-solver in the GNU Linear Programming Kit for finding transmitter settings. The policy server enforced various interference constraints that the monitors implemented, and it achieved its policy goal by changing transmitter settings of the devices it controlled. Below, we evaluate the main features of Metronome and present the key results.

A. Policy Examples

We evaluate the expressiveness and effectiveness of Metronome's policy framework under three diverse policy constraints.

Identifying interfering transmitters. Our first example policy shows how well we can make Metronome identify interference conditions and isolate the responsible interferers. The policy constraint we use to define interference is motivated in §IV-A, and specifies that Metronome should flag an interference condition when the SINR of a ZigBee or a Bluetooth receiver falls below 18 dB. To implement this policy, the policy server programmed each monitor to detect such SINR violations. SINR violations arose naturally as devices were moved around the office floor and mobile monitors sampled the environment near the receivers to compute the average signal level and the maximum interference level robustly, as described in §III-A.

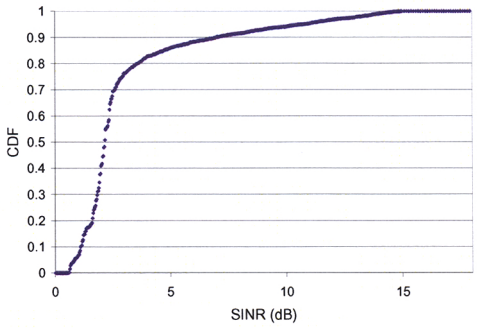

Fig. 6. CDF of SINR violations detected by the monitors.

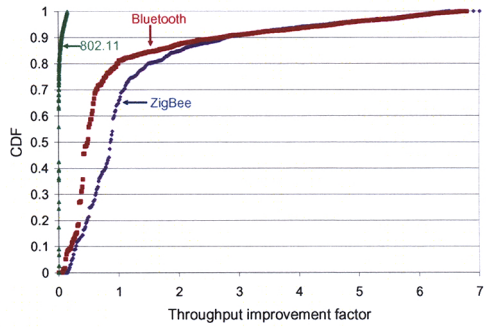

Fig. 7. Throughput improvements obtained through policy optimization.

Figure 6 shows the CDF of all detected SINR violations. The x-axis is the average SINR in dB that a monitor measures near the receiver. When the interference is higher, the SINR at the monitor decreases. The y-axis is the CDF of policy violations detected by the monitors. More than 50% of violations are due to heavy interference, which causes SINR at monitors to drop to below 3 dB. The policy server was then able to accurately isolate the interferers using the transmitter identification technique described in §III-B, which we manually verified to be sources of interference in more than 95% of the cases (external transmitters caused interference in the remaining cases).

Joint network optimization. In order to make ZigBee and Bluetooth perform well under interference, we specified a policy that separately guaranteed each ZigBee and Bluetooth receiver an SINR of 18 dB as described in §IV-A, leaving the rest of the capacity to 802.11. The difference between this policy and the previous policy is that, in this example, we want the SINR constraints at both ZigBee and Bluetooth receivers to hold simultaneously, while, in the previous example, we want to identify interferers that cause the SINR at either a ZigBee or a Bluetooth receiver to drop below 18 dB. We thus jointly optimize the performance of ZigBee, Bluetooth and 802.11 devices.

This policy resulted in the policy server selecting non-overlapping channels for ZigBee and 802.11, and in reducing the power of 802.11 transmitters just enough to respect the SINR constraints. Figure 7 shows that, under this policy, ZigBee and Bluetooth both realize a significant throughput

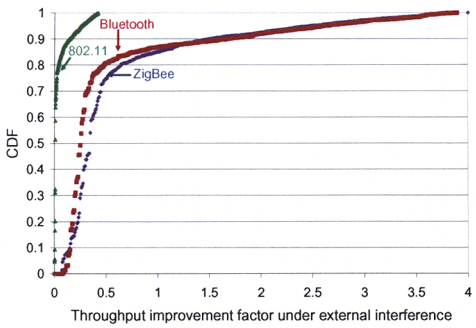

Fig. 8. Throughput improvements under a different policy for external interference.

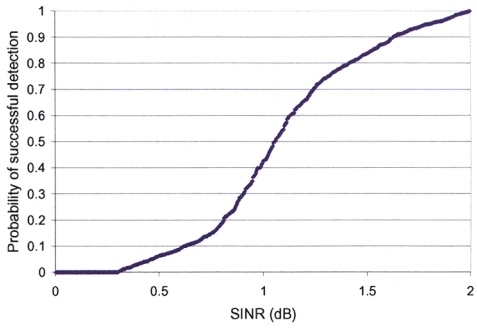

Fig. 9. Probability of successful detection at various SINR levels.

improvement ($> 6\times$), while 802.11 obtains a more modest 15% throughput gain. This outcome is because ZigBee devices were moved to a channel that doesn't overlap with 802.11, thereby reducing mutual interference between ZigBee and 802.11 and improving throughput of both networks, while 802.11's transmit power was reduced by a modest amount in order to reduce its interference on Bluetooth.

Dealing with external interference. In some scenarios, interference is caused by 802.11 transmitters that are not under the control of Metronome. We specified a different policy in order to cope with such external interference. In order not to unfairly penalize Metronome's 802.11 transmitters by causing them to reduce their transmit power when the source of interference is external, we specify SINR constraints for ZigBee, Bluetooth and 802.11 simultaneously. We assign a minimum SINR of 10 dB to 802.11 in order to keep its loss rate below 10%, while ZigBee and Bluetooth are assigned an SINR of 18 dB, as before. The policy goal is to simultaneously maximize the throughput of all three networks, by maximizing their SINRs subject to these constraints. This policy results in non-overlapping channel assignments between ZigBee and 802.11, and additionally ensures that the policy server doesn't decrease the transmit power of 802.11 devices as much as in the previous example. Figure 8 shows the throughput gains under this policy. The throughput gains of ZigBee and Bluetooth decrease to below $4\times$, but 802.11 increases its throughput to more than 40% because the policy server computes a lower interference channel for 802.11 while maintaining 802.11's transmit power. This result shows that Metronome's policies are flexible, and can incorporate fairness in the form of maintaining a maximum loss rate for each network.

B. Matched Filter Sensitivity and Extensibility

We evaluate how well the parameterized matched filter can detect activity levels, especially under low SINR conditions, and how it can implement various protocols extensibly.

We timestamped packet transmission events at senders and packet reception events at monitors. Our matched filter was able to detect packets whose SINR value as measured at the monitor exceeded 2 dB with greater than 99% probability. So, we evaluate the performance of the parameterized matched filter under low SINR conditions, which is a challenging scenario for any signal detection method [9]. Under low SINR, our parameterized matched filter sometimes fails to detect the occurrence of an event because the output of the matched filter correlator might not exceed the detection threshold. Figure 9 shows the probability of successful packet detection when the SINR was below 2 dB. The median detection rate was achieved at slightly over 1 dB, a sensitivity level that compares well with hardware-based receiver-specific designs [12, 24].

While we need more experimental evidence, our parameterized matched filter implementation demonstrates good extensibility, as shown by the fact that we were able to build matched filters for ZigBee, Bluetooth and 802.11 signals using the same hardware and software framework.

C. Metronome Cost, Scalability and Stability

Metronome is designed to be used with mobile monitors, which can cover a large region ranging from an office building to a metropolitan area using only tens of monitors. The current cost of each monitor using USRPs is less than US $1000.

We also found good scalability with regard to both bandwidth for measuring signal levels and computational costs for enforcing policy. Figure 10 shows the CDF of the bandwidth used between the policy server, monitors and transmitters under changing interference conditions. It is less than 1.5 Kbps for 12 nodes, without any protocol optimization. Also, the policy server has good computational scalability in implementing the LP-based transmitter control algorithm in §IV-B. On a standard desktop, it runs in less than five seconds for tens of devices and less than two minutes for policies simulated with hundreds of transmitters. We also found the obtained solution to be stable with respect to small variations in SINRs.

VI. Related Work

Heterogeneous wireless network sharing is a relatively new topic that has seen growing interest as the FCC has shifted toward making licensed spectrum more widely available. The two chief differences between other cognitive radios-based architectures and Metronome are that Metronome uses mobile monitors in vehicular settings, which allows us to survey a large Metropolitan area cheaply, and monitors implement parameterizable matched filtering for extensibility. We classify related work along several lines.

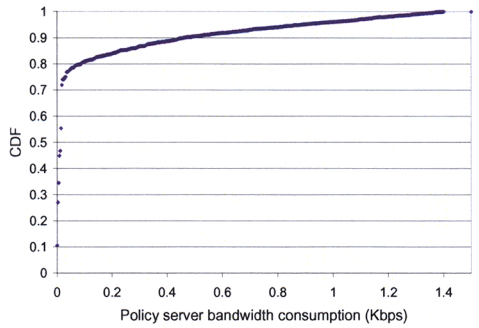

Fig. 10. CDF of the policy server bandwidth consumption.

Policy optimization in homogeneous networks. Even homogeneous wireless networks benefit from spectrum management. There are several proposals to troubleshoot 802.11 networks and reduce interference by selecting channels and power levels. These systems deploy special monitors as in DAIR [3], use laptops for troubleshooting interference and hidden terminals [26], and set AP powers [22] to minimize interference. However, while they use the same basic knobs as Metronome (power and frequency), they do not readily extend to heterogeneous wireless networks as they don't detect non-native wireless networks.

Spectrum etiquette. There have been several studies of harmful interference between dissimilar devices, e.g., 802.11 and Bluetooth, in the unlicensed bands [13, 18, 27]. Workarounds exist in specific cases, e.g., adaptive frequency hopping in Bluetooth [6], but the more general approach is spectrum etiquette [17, 19] that promote co-existence by having heterogeneous devices follow a common protocol analogous to a MAC, as first proposed early on by Bahl et al. [2]. Unlike Metronome, these schemes are combine policy and mechanism, and have little support for legacy networks such as TV receivers.

Architectures for spectrum sharing. Several architectures for coordinating spectrum sharing have been proposed in the areas of dynamic spectrum access and cognitive radios; Akyildiz et al. [1] provide a survey of recent work in the area.

VII. Conclusions

We presented Metronome, an extensible architecture for coordinating spectrum sharing between heterogeneous networks in both licensed and unlicensed bands. Metronome provides a flexible and expressive policy language for specifying SINR constraints and goals, and separates this policy from a small set of core mechanisms. These mechanisms include continuously checking for SINR violations at the monitors, isolating the offending interferers, and recomputing transmission parameters so as to meet the policy goals and constraints. It exploits monitor mobility to scale up to a large metropolitan area while maintaining measurement accuracy of signal and interference levels experienced at receivers. It uses parameterized matched filtering to separate signal levels from interference levels, and hence detects problematic interference scenarios better than energy detection. It reconfigures the transmitter settings of devices it controls, and hence improves their throughput by reducing mutual interference and interference from external transmitters. In our experiments on an indoor 2.4 GHz testbed, Metronome improves the throughput of ZigBee and Bluetooth by more than 6× without degrading competing 802.11b/g by computing better transmitter parameters.

Acknowledgments. We thank David Wetherall for his thoughtful comments during the early phase of this work.

References

[1] I. F. Akyildiz, W. Y. Lee, M. Vuran, and S. Mohanty. NeXt Generation / dynamic spectrum access / cognitive radio wireless networks: A survey.
[2] V. Bahl. Spectrum etiquettes for short range wireless devices operating in the unlicensed band - a proposal.
[3] V. Bahl, R. Chandra et al. Enhancing the security of corporate WiFi networks using DAIR. In *MobiSys 2007*.
[4] BBN 802.11 GNURadio implementation,http://tinyurl.com/34gv47.
[5] Bluesniff Bluetooth GNURadio implementation, http://bluesniff.shmoo.com/.
[6] Bluetooth AFH, http://tinyurl.com/2grndv.
[7] V. Brik, E. Rozner, S. Banarjee, and P. Bahl. DSAP: A protocol for coordinated spectrum access. In *DySPAN 2005*.
[8] M. M. Buddhikot, P. Kolodzy, S. Miller, K. Ryan, and J. Evans. DIMSUMNet: New directions in wireless networking using coordinated dynamic spectrum access. In *IEEE WoWMoM, 2005*.
[9] D. Cabric, S. M. Mishra, and R. W. Brodersen. Implementation issues in spectrum sensing for cognitive radios. In *Asilomar 2004*.
[10] V. Chvatal. *Linear Programming*. W. H. Freeman, 1983.
[11] Competitive Bidding Procedures for Auction of 700 MHz Licenses, http://tinyurl.com/3cf7n5.
[12] A. Goldsmith. *Wireless Communications*. Cambridge University Press, 2005.
[13] Home Wi-Fi Network Owners and Interference, by Jupiter Research.
[14] IEEE 802.11y task group, http://tinyurl.com/2m6zop.
[15] Initial Evaluation of the Performance of Prototype TV Band White Space Devices, http://tinyurl.com/yp7fmw.
[16] K. Jamieson and H. Balakrishnan. PPR: Partial packet recovery for wireless networks. In *SIGCOMM '07*.
[17] X. Jing and D. Raychaudhuri. Spectrum coexistence of IEEE 802.11b and 802.16a networks using CSCC etiquette protocol. In *DySPAN 2005*.
[18] A. Kamerman. Coexistence between Bluetooth and IEEE 802.11 CCK : Solutions to avoid mutual interference.
[19] J. Lee, J. Mo, T. M. Trung, J. Walrand, and W. So. Wiflex: Multi-channel cooperative protocols for heterogeneous wireless devices.
[20] L. Ma, X. Han, and C.-C. Shen. Dynamic open spectrum sharing MAC protocol for wireless ad hoc networks. In *DySPAN 2005*.
[21] M. McHenry. Frequency agile spectrum access techniques. In *FCC workshop on cognitive radio*, 2003.
[22] K. Mhatre, D. Papagiannaki, and F. Baccelli. Interference mitigation through power control in high density 802.11 WLANs. In *INFOCOM 2007*.
[23] S. Narlanka, R. Chandra, P. Bahl, and J. I. Ferrell. A hardware platform for utilizing the TV bands with a Wi-Fi radio. In *IEEE LANMAN 2007*.
[24] J. G. Proakis. *Digital Communications*. Mc Graw-Hill, 2000.
[25] S. Sankaranarayanan, P. Papadimitratos, and A. Mishra. A bandwidth sharing approach to improve licensed spectrum utilization. In *DySPAN 2005*.
[26] A. Sheth, C. Doerr, D. Grunwald, R. Han, and D. Sicker. MOJO: a distributed physical layer anomaly detection system for 802.11 WLANs. In *MobiSys 2006*.
[27] The effects of interference on general WLAN traffic. Document FPG 2006-328.2, by Farpoint Group.
[28] UCLA ZigBee GNURadio implementation, http://tinyurl.com/3cmrp8.
[29] Y. Yuan, P. Bahl et al. KNOWS: Kognitiv networking over white spaces. In *DySPAN 2007*.