Wireless Networks Should Spread Spectrum Based On Demands *

Ramakrishna Gummadi   Hari Balakrishnan
MIT CSAIL

ABSTRACT

Today's local-area, mesh and cellular networks assign a single narrow-band channel to a node, and this assignment remains fixed over long time scales. Using network traces, we show that the load within a network can vary significantly even over short time scales on the order of tens of seconds. Therefore, we make the case for allocating spectrum *on-demand* to nodes and regions of the network that need it. We present an architecture that shares the entire spectrum on-demand using spread-spectrum codes. If implemented, the system will achieve fine-grained spectrum allocation for bursty traffic without requiring inter-cell coordination. Preliminary experiments suggest a throughput improvement of 75% over commodity 802.11b networks. By eschewing the notion of channelization, and matching demand bursts with spectrum dynamically, better wireless networks that sustain higher throughputs may be designed.

1   INTRODUCTION

Wireless spectrum is a precious resource. The holy grail for the designers of wireless data networks is to maximize the aggregate network throughput within the frequency band alloted to the network. The current approach toward this goal is to first provision frequency bands to access points that form cells. Then, within each cell, a MAC protocol determines which nodes in the cell can transmit at any given time. Together, provisioning and scheduling attempt to ensure high spatial reuse (i.e., maximize the number of successful concurrent transmissions), thereby improving throughput.

In most current in-building or campus-wide wireless LANs, network administrators provision cellular resources over long time scales (weeks or months). Even in situations where access points (APs) are able to dynamically pick their operating channels from a wide-band selection, they pick a fixed-width channel in which to operate. The result is that an AP or a cellular base station uses a fixed chunk of the spectrum whenever it transmits, as does a client within a given cell. This fixed-width allocation causes significant throughput problems due to congestion on the wireless medium. Such problems have been identified at individual 802.11 hotspots [15] as well as at sites with multiple APs [11].

Fundamentally, a fixed spectrum allocation is sub-optimal because it does not track demand, which varies across different regions in the network and with time. Prior work has reported significantly varying demands at conferences [3],

on campuses [12], and in enterprises [5]. For example, consider a conference hotel that runs multiple APs, each assigned a different channel to reduce interference. During the day, spectrum resources ought to be allocated to the APs in the conference rooms and away from the APs where there are few users. This strategy would achieve higher network throughput. The same argument applies to a typical office building, or to the wide-area cellular system during rush hour, or to disaster relief situations when many agencies and nodes all converge at a given location. 802.11 and cellular networks use a variety of techniques such as Dynamic Frequency Selection (DFS) [10] and cell breathing [2] to distribute load across cells. These technologies shift demand to cells that are lightly loaded, but weaken the received signal strength and limit throughput. In contrast, we advocate moving spectrum to cells that see higher demands.

We are not the first to recognize this fundamental shortcoming in existing wireless networks. A recent paper by Moscibroda et al. [14] makes the case for replacing the fixed-width channel of an AP and its client with a variable-width channel that is adjusted at ten-minute intervals in order to capture demand variations seen across APs. This proposal is sufficient if the observed demand at an AP is roughly constant over the channel-width update period. However, if the traffic is bursty, it can waste spectrum because nodes using narrow-width channels that have data to send cannot use the spectrum allocated to temporarily idle cells assigned broader-width channels. Decreasing the channel-width update period increases inter-AP coordination and can decrease stability and induce oscillations. In this paper, we focus on the problem of improving the throughput of networks with variable demands that are also highly bursty.

First, we present a trace study of wireless packet traces from the OSDI 2006 conference showing that demands can be both highly variable and bursty. Then, we argue that, for such networks, we should dispense with the notion of channelization. To demonstrate the validity of this viewpoint, we design and implement a direct-sequence spread-spectrum architecture called ODS (On-Demand Spectrum), in which every node uses the entire available spectrum. A node spreads its signals across the entire spectrum using a fixed-length pseudo-random spreading code. Such spreading decreases the received throughput compared to a non-spreading transmitter that uses the entire spectrum, even if the spectrum is large enough to accommodate the bandwidth expansion incurred by spreading. To compensate for this throughput loss, we allow a node to use more than one spreading code at

the same time and bond its transmissions. The exact number of codes a node uses simultaneously depends on its demand, as well as on the demands of other interfering nodes. If no interfering node has data to send, the node increases the number of spreading codes it uses to recover the throughput loss incurred by spreading. This policy decreases the effective spreading factor and simulates a non-spreading transmitter that uses the entire spectrum, not just a single fixed-width channel. If some interfering nodes have data to send, the node decreases the number of codes it uses by a corresponding amount.

ODS uses a random policy for selecting the pseudo-random spreading codes, and an adaptive receiver feedback mechanism to handle the challenging problem of fine-grained spectrum allocation without requiring excessive synchronization. Although the idea of using spread-spectrum to reduce synchronization while sharing spectrum is not new, what is new is the mechanism to regulate the number of spreading codes based on observed demand. In the process, ODS resolves the tension between the main appealing aspect of CSMA, which is that any node can send data on an entire channel without delay or coordination, and the chief benefit of channelization, which is that a network operator can limit interference across spatial regions.

We prototyped ODS using the USRP software radio platform. Our system allows us to transmit signals at 1 and 2 Mbps data rates, which are spread using a 11-chip pseudo-random spreading code similar to the Barker code used in 802.11b. Since the USRP is too slow to sustain the high data rates of the chipped spread-spectrum signals, we implement the spread-spectrum functionality within the USRP FPGA, and transmit only the low-rate data streams between the USRP and the host PC. Our implementation is preliminary and unoptimized. We compare the performance of ODS against commodity 802.11b radios that use a fixed Barker code in every device. Even when six transmissions interfere, we find that, where 802.11b achieves only 45% of the aggregate throughput without interference, ODS achieves 80% of aggregate throughput, an improvement of 75% over 802.11b. This improvement results from assigning spectrum using multiple codes based on demand, instead of using a single and fixed spreading code.

2   THE CASE FOR SPECTRUM ON DEMAND

To demonstrate that traffic can be both highly variable and bursty, we present a small set of results obtained from over-the-air traces collected during the OSDI 2006 conference [4]. Several recent studies have examined the performance of 802.11 networks in hotspot settings extensively [11, 15], and found that such networks perform worse than expected under congestion because of contention-related losses and delays, as well as subsequent retransmissions and rate fall-backs. These studies also suggest that the best way to improve performance is to send smaller packets at faster rates.

In contrast, we posit that the primary contributor to such poor performance is the varying nature of the demand itself over both short (tens of seconds) and long (minutes to hours) time scales. From the OSDI packet traces, we were able to easily identify short time-periods (30-second intervals) in which the demand across various APs varied by more than a factor of 3.5. Further, we found that demands can change completely over long time scales of several minutes to hours when factors such as user movement and activity cause demand to be shifted to a different portion of the network.
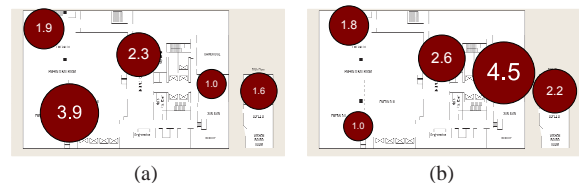

Figure 1: Relative demands across five APs during two consecutive 30-second intervals (the areas of the circles are proportional to APs' demands). Demand can thus be both widely variable and bursty.

Figure 1 shows the traffic demands at five APs over two consecutive 30-second intervals. An AP's demand is calculated as the amount of data originated by the AP within the 30-second period as determined by all sniffers that were able to observe the AP. Even though the APs were well-engineered in terms of orthogonal channel assignments and placements, Figure 1 shows that demands are highly variable and bursty. While Figure 1 shows only one data point of bursty demands, we have found that instances of such bursty patterns occur frequently in the trace. We leave a thorough quantification of demand variability to future work.

3   ON-DEMAND SPECTRUM ARCHITECTURE

ODS makes two major changes to the way spectrum is currently allocated and used:

1. ODS allocates spectrum to nodes dynamically based on their demands, and

2. ODS enables nodes to exploit concurrent transmissions by allocating multiple spreading codes to nodes.

ODS has three components. The first one is a mechanism that allows a receiver to estimate the future traffic demands of its transmitters so that it can allocate the spectrum across these transmitters. This mechanism works over short time scales of several packet transmissions. The second is a mechanism for receivers to decide how to assign spreading codes to transmitters according to the estimated demands. The third is a mechanism to ensure that concurrent transmissions using these codes can occur successfully, by allowing a transmitter to adaptively discover how many of its allocated codes can be successfully used before mutual interference due to concurrent transmissions decreases usable capacity.

We first describe the mechanisms that allow a receiver to determine the traffic demands of transmitters, and use them

to estimate the transmitters' code allocation (§3.1). Because it is infeasible to coordinate receivers during code allocation, we propose and analyze the performance of a random code-selection policy that assigns a fixed-length pseudo-random number (PN) code sequences to transmitters in an uncoordinated manner. We fix the length of the PN codes at 11 chips, for 802.11b compatibility. We show that the random code-selection policy has good expected performance, while its best-case performance approaches that of the optimum centralized assignment to within a constant factor of e (§3.2). Then, we describe how a transmitter uses all its codes concurrently (§3.3), and finally describe how transmitters adaptively detect when excessive concurrency turns into interference (§3.4).

3.1 Code Allocation

We assume that each node n has some packet transmission demand of d_n bits that must be transmitted as soon as possible, and can transmit at an average rate of r_n bits/s. r_n is the average bit-rate that the transmitter sees after rate-adaptation, which works at smaller time scales than demand scheduling. We assume that there are enough codes in the system, and that every node has access to at least one code by default. We assume that these codes can be decorrelated well at a receiver. The code availability and the decorrelation assumptions can be approximated in reality by using PN sequences that have low cross-correlation.

ODS allocates PN codes to transmitters in proportion to their demands. Demands are dictated both by the actual number of bits d_n that a transmitter needs to send, and the average bit rate r_n at which it can send them. Each receiver adaptively estimates these quantities on behalf of its transmitters, based on previously observed demands and rates of the transmitter. This estimation procedure is a simple moving average filter over a period of 30 seconds, which we found works well on the OSDI traces.

Once the receiver estimates its transmitters' demands, it assigns each transmitter n a number of codes $c_n = \left\lceil c \frac{\frac{d_n}{r_n}}{\sum_i \frac{d_i}{r_i}} \right\rceil$, where c is the codebook size, which is the total number of available codes. ODS uses a codebook size of $c = 128$ by default, which is large enough to utilize a 22 MHz-wide 802.11b spectrum fully. Further, this code assignment means that, assuming that the number of clients associated with an AP is not more than $c = 128$, every node gets at least one PN code (which is statically configured).

Interestingly, yet somewhat counter-intuitively, it follows from this formula that, given two transmitters with the same data load but different average bit rates, a receiver allocates more codes to the slower transmitter that to the faster transmitter, so as to increase concurrency, and improve the mean packet transmission time. Such a policy has fairness implications different from the status quo, and we defer a thorough study to future work.

Two potential issues in ODS are security (including various forms of Denial of Service concerns) and mobility. Since codes are ultimately allocated by the receiver, it is possible to enforce expressive policies for a transmitter's code allocation at the receiver. Further, since every receiver has access to at least one PN code, it is not possible for selfish nodes to completely deny network access, because spread-spectrum provides some jamming immunity as long as the statically assigned code is kept secret from the jammer. Small amounts of mobility do not pose serious problems to ODS because a receiver dynamically allocates codes to transmitters on a short-term basis, and because each node's statically assigned code is portable. However, continued mobility could cause problems, and we defer this problem to future work.

3.2 Code Selection

ODS uses an uncoordinated code assignment policy based on random selection of PN codes. Each receiver assigns a certain number of randomly chosen codes from a relatively large, but fixed, codebook of PN codes to each of its transmitters, without coordinating with other receivers. Conflicts may arise in such a receiver-driven code selection when two uncoordinated receivers allocate the same PN code to their transmitters. We assume that when two concurrent transmissions use the same code, they are both corrupted. Otherwise, both transmissions are correctly decoded. This is a pessimistic model because, depending on the received signal strengths, one or both transmissions might still be successfully decoded.

We now analyze the throughput of this code-selection policy. Let k denote the number of randomly selected codes assigned to each transmitter T, and let n denote the number of receivers around T. From the perspective of T, the expected number of conflict-free codes λ it expects to be able to select is $\lambda = k(1 - \frac{k}{c})^n$. The reason is that each of the k codes has a probability of $1 - \frac{k}{c}$ of not being in conflict with the code selected by any other receiver in T's vicinity. Due to the independence selection property of codes and concurrent transmitters, this formula for λ captures the expected number of conflict-free codes selected by this policy.

λ represents the expected throughput achievable using conflict-free codes, not including the one code statically assigned to every node. In Figure 2, we plot the performance of λ as we increase the number of codes k allocated to each node. The number of available codes $c = 128$. Each curve in the plot represents the average throughput improvement seen by a transmitter using multiple codes over a transmitter using a single code, when other contending nodes also pick k codes independently.

We show that random code-selection is both efficient and robust. For a given number of contending users n and a given code size c, the per-node throughput $\lambda = k(1 - \frac{k}{c})^n$ is optimized when $k_{\text{opt}} = \frac{c}{n+1}$, and is equal to $\lambda = \frac{c}{n+1}(\frac{n}{n+1})^n$. As we increase both the code size and the number of contending nodes keeping their ratio fixed, λ asymptotically approaches $\frac{c}{ne}$. Thus, the optimum uncoordinated random code selection is within a constant factor of the optimum fully coordinated
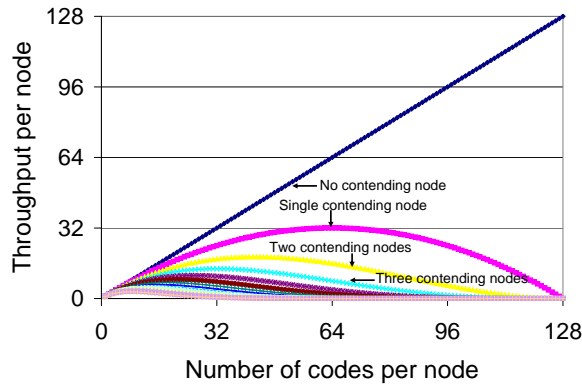
Figure 2: Per-node throughput under varying number of interferers.

strategy. Further, from the shape of the curves in Figure 2, it is apparent that the penalty for sub-optimal selection of k is not severe as long as k is chosen to be approximately equal to k_{opt}. Thus, random selection is robust to incorrect estimation of the number of contending users n.
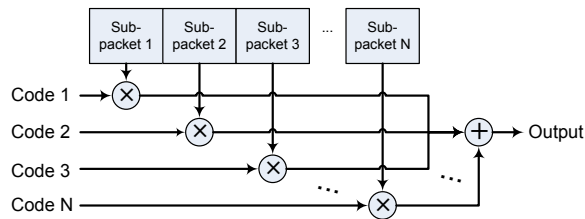
3.3 Code Bonding

Figure 3: Illustration of bonding. Each sub-packet is spread using a different PN code, and all sub-packets are sent in parallel.

Bonding is a way of sending multiple sub-packets of a packet concurrently using separate PN codes for each of these sub-packets (Figure 3). The motivation is that, during intervals of low demand, a large portion of the entire spectrum can be allocated to a single node, which can then use it to speed up the overall packet transmission by first spreading the individual sub-packets with their own spreading codes and then multiplexing the coded signals onto the wide-band spectrum (Figure 3). Similarly, during intervals of high demand, fewer codes can be allocated per node, so that fewer sub-packets can be bonded; in the worst-case, every node uses only one code, so that there is no sub-packet bonding. The bonding limit is dictated by SINR considerations. As the number of sub-packets increase, the coding distance between two coded signals decreases, so it makes it harder for the receiver to accurately decorrelate the sub-packets in the presence of interference and noise.

3.4 Feedback-based Adaptation

ODS makes the entire spectrum usable by one or more nodes, so we aim to maximize concurrent transmissions, as long

as they do not cause unacceptable interference to other concurrent transmissions. Statically deciding what the optimum amount of concurrency is for an arbitrary topology is an extremely challenging problem: if an active transmitter uses too few codes, spectrum is wasted, but if too many transmitters use too many codes concurrently, the achieved capacity is decreased because every code of every transmitter interferes with other transmitters.

To safely bound concurrency, ODS uses an adaptive mechanism that uses feedback from the receiver. A transmitter assumes that a coded transmission is lost due to mutual interference with some probability p. If it is correctly received, the transmitter has overestimated the interference from other concurrent transmitters, and so decreases p. If it is incorrectly received, the transmitter increases p. In the ideal case when the channel and other traffic are both invariant, the probability will converge to either 1 or 0, depending on the presence or absence of mutual interference. If the probability reaches 1, the transmitter decreases the code rate by dropping that code from its allocated code set and decreasing its coding rate. So, the transmitter's own performance improves because of the lowered coding rate, which also has the positive effect of simultaneously reducing the network-wide mutual interference levels. In case network conditions improve, the transmitter receives positive feedback about this fact from the receiver's decoder, which will see improved decoding performance. On the other hand, if conditions deteriorate, the transmitter will decrease the coding rate. We defer a careful study of protocol dynamics such as the adaptation rate and stability to future work.

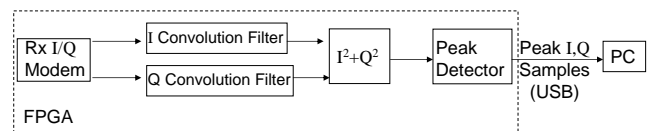## 4 IMPLEMENTATION AND EVALUATION

Figure 4: ODS PN-code despreader design.

We built an ODS prototype using the USRP hardware. Our main challenge was implementing high-rate coded samples. The current USRP is limited by the throughput of the USB bus to a throughput of 32 MB/s. Supporting an 802.11b-compliant spread-spectrum stream means, assuming 2 Mbps data and 11-chip codes, we must support $2 \times 11 \times 2 = 44$ Msps to satisfy Nyquist sampling. Since each sample is 16-bits, we need a throughput of 88 MB/s, which cannot be met. Instead, we implemented support for spreading and despreading the data in the FPGA on the USRP itself, so that only the actual data needs to be shipped across USB.

This design is shown in Figure 4, which shows the ODS-specific signal processing that is carried out on the FPGA for the receiver section; transmitter section is similar. The incoming I, Q samples are decorrelated with the ODS-selected

spreading code in the "Convolution Filter" blocks. We then sum the amplitudes of the filtered I, Q samples and look for peaks in the summed signal. We output only these peak samples, which correspond to the decorrelated data values. Our implementation of random coding was based on the Barker receiver implementation provided by the Utah SPAN lab [6].
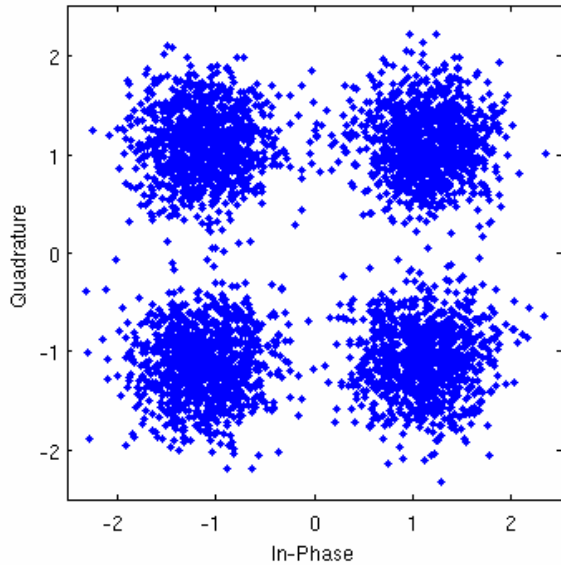

Figure 5: I,Q outputs of the PN-code despreader.

Figure 5 shows the decorrelated values of the I, Q symbols as received on the PC for a 2Mbps DQPSK transmission. The symbols are clustered depending on what 2-bit data values they were modulated with. Thus, the spreading/despreading implementation provides satisfactory performance. We then do data demodulation, as well as ODS-specific processing, such as code allocation and multi-code bonding, on the host PC. Since the FPGA can only support only one PN-code despreading, we use multiple USRPs to implement bonding.

To test the end-to-end performance of ODS, we show the BER (bit-error rate) plots of the received data at varying SINR (signal-to-interference-plus noise) ratios with and without interference. We calibrate received and noise powers using a spectrum analyzer. Figure 6 shows the BER vs. SINR of a receiver with and without interference. Data can be received at relatively low SINRs because of the spread-spectrum processing gain. Further, the throughput does not degrade significantly with interference because the two concurrent transmissions use randomly selected PN codes.

To test ODS under different demands and levels of interference, we used a configuration with twelve nodes and six interfering links. We measured the throughput obtained on a link that could bond two 802.11b channels to obtain up to 4 Mbps without interference. We then increased the interference (and, hence, demands) on other links, and measured the bonded link's throughput when the number of in-
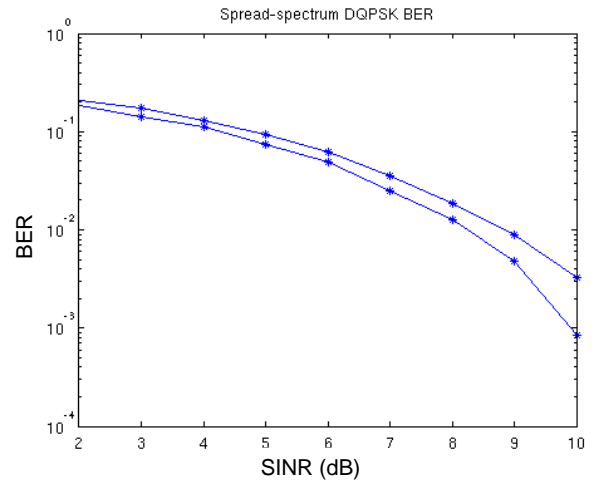

Figure 6: SINR vs. BER plots with and without interference.

terfering links is varied between 1 and 5. Our main finding is that, even in this high interference scenario and with a relatively unoptimized implementation, ODS could sustain up to 80% of the ideal throughput of 4 Mbps (i.e, it achieved 3.18 Mbps) across all six links, while, under similar conditions, 802.11b PRISM cards could only manage 1.8 Mbps in total. Thus, ODS improves 802.11b throughput by more than 75% by tolerating interference better under loaded conditions. We leave large-scale experiments to future work.

5 RELATED WORK

ODS uses spread-spectrum codes instead of frequencies. There are both pros and cons with this choice. While frequency-division multiplexing can provide orthogonality without causing mutual interference, it suffers from a significant drawback within our architectural context—codes can be finely divided and allocated, but fine-grained frequency division requires more sophisticated hardware than is currently available. For example, even though a current wireless chipset such as Atheros 5005GS can support variable-width channels of 5, 10 and 20 MHz [14] and bit-rates down to 0.25 Mbps, it still consumes a 5 MHz spectrum at a minimum to support the 0.25 Mbps rate. In contrast, commodity PRISM chips such as HSP3824 [9] provide a 16-chip spread-spectrum programmability. However, the advantage of using frequencies is that we can use much higher bit-rates with commodity cards (up to 54 Mbps with OFDM modulation used in 802.11a/g). In an accompanying paper [8], we exploit this high bit-rate facility along with variable-width channel support to study how much throughput improvements are obtainable with non-bursty, backlogged flows.

Spread-spectrum codes are used widely in cellular networks. For example, IS-95 voice networks and CDMA2000 data networks use spread-spectrum codes. However, these systems allocate a fixed amount of spectrum to a base station, and a heavily-loaded base station cannot borrow spectrum from its neighboring cells (which are on different chan-

nels, in order to mitigate co-channel interference). Instead, users are redirected to neighboring cells, which means the received signal is weaker than if spectrum were allocated locally based on demand. ODS can be applied to such cellular networks to handle bursty traffic. A heavily loaded ODS base station allocates more codes to its clients, while a lightly loaded base station apportions fewer codes.

CDMA has been proposed as a solution for resource multiplexing in a number of previous proposals for multi-hop wireless networks (e.g., [16]). The basic idea is that, by using spread-spectrum processing, geographically separated nodes can communicate concurrently in a much more efficient manner than CSMA would allow. Each node in the network is assigned a single code. Their main goal is not to deal with variable demands but to maintain communication links among neighbors under disruptions due to interference or mobility, by carefully reducing the transmit power of some radios in the network. In contrast, ODS does not alter the transmit power of nodes. Instead, it allocates more bonded codes to nodes that have higher demands, either because they actually have more data to send or because they are connected at lower bit rates than other nodes due to interference or noise. Our overall goal of satisfying short-term demands by allocating more spectrum to more demanding portions of the network is also different than the disruption-tolerance focus of these works.

At the link level, several frequency hopping strategies to mitigate interference have been proposed recently, e.g., SSCH [1] and MAXchop [13]. Some commercial APs also switch frequencies dynamically to minimize external interference. All these works still assume constant and equal internal demand at all nodes, while ODS allocates the entire spectrum based on fluctuating traffic demands at nodes.

6 CONCLUSIONS AND FUTURE WORK

We made the case for handling bursty traffic better in wireless networks. We presented ODS, which achieves uncoordinated and fine-grained spectrum allocation based on observed demands. We found that ODS improves the throughput of interfering links by 75% over 802.11b under high interference conditions. Our preliminary results suggest that wireless networks can see significant throughput improvements by eschewing channelization completely, and instead by matching bursty demands with spectrum dynamically.

We plan to conduct a more thorough evaluation of ODS at higher bit-rates and larger topologies. More fundamentally, we would like to characterize the capacity and achievable rates of wireless networks with bursty traffic. Unlike the traditional notion of Shannon capacity that determines the fixed rates achievable by nodes, the instantaneous throughput with bursty traffic depends on the total received power, which in turn depends on the number of active transmitters [7]. So, we want to characterize this new notion of "bursty capacity" as a function of nodes' duty cycles and received powers.

REFERENCES

[1] P. Bahl, R. Chandra, and J. Dunagan. SSCH: Slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks. In *MobiCom'04*.

[2] P. Bahl, M. T. Hajiaghayi et al. Cell breathing in wireless LANs: Algorithms and evaluation. *IEEE TMC*, 6 (2), 2007.

[3] A. Balachandran, G. M. Voelker, P. Bahl, and P. V. Rangan. Characterizing user behavior and network performance in a public wireless lan. In *SIGMETRICS'02*.

[4] R. Chandra, R. Mahajan, and V. Padmanabhan. OSDI'06 traces. URL `ftp://ftp.research.microsoft.com/pub/ratul/OSDI2006-data/`.

[5] Y.-C. Cheng, J. Bellardo et al. Jigsaw: Solving the puzzle of enterprise 802.11 analysis. In *SIGCOMM'06*.

[6] Full-bandwidth 802.11b implementation, `http://span.ece.utah.edu/pmwiki/pmwiki.php?n=Main.80211bReceiver`.

[7] R. Gallager. A perspective on multiaccess channels. *IEEE Transactions on Information Theory*, 31(2), Mar 1985.

[8] R. Gummadi, R. Patra, H. Balakrishnan, and E. Brewer. Interference avoidance and control. In *HotNets'08*.

[9] HSP3824 Direct-Sequence Spread-Spectrum Baseband Processor Data Sheet, `http://www.datasheetcatalog.org/datasheets2/28/283233_1.pdf`.

[10] IEEE 802.11h: Spectrum managed 802.11a (5 GHz) for European compatibility (2004), `http://ieeexplore.ieee.org/xpls/abs_all.jsp?tp=&isnumber=27870&arnumber=1243739&punumber=8810`.

[11] A. P. Jardosh, K. N. Ramachandran, K. C. Almeroth, and E. M. Belding-Royer. Understanding link-layer behavior in highly congested IEEE 802.11b wireless networks. In *E-WIND'05*.

[12] D. Kotz and K. Essien. Analysis of a campus-wide wireless network. In *MobiCom'02*.

[13] A. Mishra, V. Shrivastava, D. Agrawal, S. Banerjee, and S. Ganguly. Distributed channel management in uncoordinated wireless environments. In *MobiCom'06*.

[14] T. Moscibroda, R. Chandra, Y. Wu, S. Sengupta, and P. Bahl. Load-aware spectrum distribution in wireless LANs. In *ICNP'08*.

[15] M. Rodrig, C. Reis, R. Mahajan, D. Wetherall, and J. Zahorjan. Measurement-based characterization of 802.11 in a hotspot setting. In *E-WIND'05*.

[16] T. J. Shepard. A channel access scheme for large dense packet radio networks. In *SIGCOMM'96*.