Cross-Layer Wireless Bit Rate Adaptation

Mythili Vutukuru and Hari Balakrishnan
MIT CSAIL
{mythili,hari}@csail.mit.edu

Kyle Jamieson
University College London
k.jamieson@cs.ucl.ac.uk

ABSTRACT

This paper presents *SoftRate*, a wireless bit rate adaptation protocol that is responsive to rapidly varying channel conditions. Unlike previous work that uses either frame receptions or signal-to-noise ratio (SNR) estimates to select bit rates, SoftRate uses confidence information calculated by the physical layer and exported to higher layers via the *SoftPHY interface* to estimate the prevailing channel bit error rate (BER). Senders use this BER estimate, calculated over each received packet (even when the packet has no bit errors), to pick good bit rates. SoftRate's novel BER computation works across different wireless environments and hardware without requiring any retraining. SoftRate also uses abrupt changes in the BER estimate to identify interference, enabling it to reduce the bit rate only in response to channel errors caused by attenuation or fading. Our experiments conducted using a software radio prototype show that SoftRate achieves 2× higher throughput than popular frame-level protocols such as SampleRate [4] and RRAA [24]. It also achieves 20% more throughput than an SNR-based protocol trained on the operating environment, and up to 4× higher throughput than an untrained SNR-based protocol. The throughput gains using SoftRate stem from its ability to react to channel variations within a single packet-time and its robustness to collision losses.

Categories and Subject Descriptors: C.2.1 [Computer-Communication Networks]: Network Architecture and Design—Wireless communication

General Terms: Design, experimentation, performance.

Keywords: Wireless, bit rate adaptation, SoftPHY, cross-layer.

1. INTRODUCTION

Wireless communication suffers from many time-varying vagaries that cause bit errors and packet losses. These include signal *attenuation*, channel *fading* due to multipath propagation, and *interference* caused by other transmissions at overlapping frequencies. These stochastic effects are more pronounced when changes occur in the propagation environment, for instance because of node mobility, or by the movement of people and objects. The result is a channel that is difficult (if not near-impossible) to accurately model, in which the signal-to-noise ratio (SNR) and channel bit error rate (BER) change with time. For example, Figure 1 shows measurements that illustrate the variation of SNR and BER over time when a sender is moving away from the receiver at walking speed; note the multipath fading effects on shorter timescales in addition to the gradual attenuation over longer timescales.

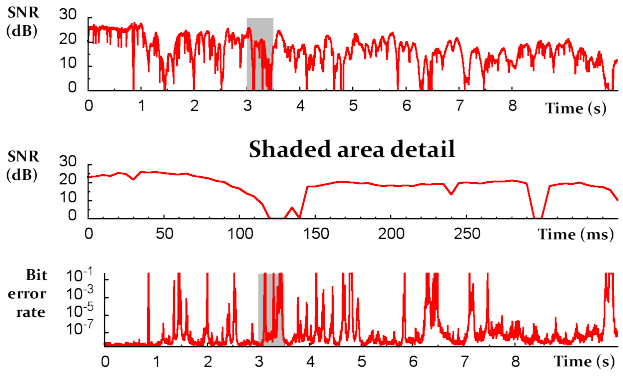


Figure 1: Experimental SNR fluctuations in time over a fading channel with walking-speed mobility. Large-scale fading is evident from the 10-second window (upper), and in a 350 ms detail (middle) we see fades a few tens of milliseconds in duration. Bit error rate (lower: BPSK, code rate-1/2) changes with SNR. Data obtained using an 802.11a/g-like software radio prototype (§4).

To improve throughput in these varying conditions, the sending node can dynamically adapt its modulation and coding by picking a suitable *bit rate*. The *bit rate adaptation protocol* used to make this choice must answer two important questions:

1. What signal (information) should the sender use to select the right bit rate?
2. Over what timescale should this signal be observed?

Prior work on bit rate adaptation (§2) uses one of two information signals: frame receptions or signal-to-noise ratio (SNR). Frame-level protocols [24, 4] must operate over the timescale of tens or hundreds of frames or more because they need several transmissions to accurately assess frame loss rates at various bit rates. As a result, frame-level schemes are not responsive to channel variations that occur on shorter timescales. On the other hand, SNR-based protocols [10, 21] can operate on shorter timescales by estimating the SNR on each reception and mapping it to the expected BER using known SNR-BER relationships. But because the BER at a given SNR might vary by many orders of magnitude between environments, these protocols must be carefully trained for each

SIGCOMM'09, August 17–21, 2009, Barcelona, Spain.

operating environment [5]. SNR measurements also require hardware-specific calibration [25].

The information signal used by rate adaptation protocols must also be robust to interference. A bit rate adaptation protocol must not reduce bit rate in response to collisions, because doing so increases the transmit duration of frames and conflicts with other mechanisms (like exponential backoff) that the channel access protocol employs to avoid a collision on the next retry. A frame reception is an example of an information signal that is not robust to interference [24, 20].

This paper presents *SoftRate*, a bit rate adaptation protocol that overcomes these limitations of existing protocols. SoftRate uses a novel signal to make its decisions: *the interference-free BER estimate* computed using per-bit confidences exported by the physical layer (PHY). Note that these per-bit confidences, usually referred to as *SoftPHY hints* [12], were computed only for the Zigbee PHY in previous work. In this paper, we generalize the concept of SoftPHY hints and show how one can compute them for any PHY (including 802.11a/b/g, Zigbee, WiMax) that uses a linear convolutional or block error-correcting code. We propose using the *log-likelihood ratio* of a bit being correct to its being incorrect that is computed by some standard decoders [8, 2] as the SoftPHY hint, and show that it can be used to accurately estimate the underlying channel BER.

A SoftRate receiver uses the per-bit SoftPHY hints delivered by the PHY via the SoftPHY interface to accurately estimate the BER of a received frame without knowing which bits were actually transmitted. Furthermore, our method allows the receiver to estimate the underlying channel BER even using a frame that was received with *no* errors, a feature that is important in the context of bit rate adaptation (e.g., channel BER estimates of 10^{-4} and 10^{-9} at some bit rate would result in different transmit bit rate choices for the next packet). The SoftRate receiver also uses a heuristic to separate out errors caused by strong interferers, because reducing the transmit bit rate in response to interference only worsens the contention on the channel. The SoftRate sender then uses the interference-free BER conveyed by the receiver at the current bit rate to estimate the BER at the other rates, and before each transmission picks the bit rate that minimizes the air-time required to deliver the packet to the receiver (§3). Using a very small amount of information on the feedback channel—one BER measurement per frame—SoftRate adapts the transmit bit rate *at the granularity of individual frames*, and is highly responsive to rapid channel variations due to mobility.

We have implemented our SoftPHY scheme using minor modifications to the 802.11a/g-like PHY in the GNU Radio codebase (§4). Experiments with our software radio prototype show that SoftPHY hints can be used to correctly estimate packet BER without requiring any training or calibration across a wide variety of wireless propagation environments (§5). Our trace-driven evaluation of TCP over SoftRate using the ns-3 simulator (§6) shows that SoftRate achieves gains of 20% over an SNR-based protocol carefully trained on the operating environment, $4\times$ higher throughput than an untrained SNR-based protocol, and up to $2\times$ more throughput than frame-level protocols like RRAA [24] and SampleRate [4] in mobile fading and interference-dominated channels. Performance gains in our experiments stem from SoftRate's ability to quickly react to rapid channel variations before TCP's end-to-end congestion control mechanism reacts to burst losses, and its resilience to collision-induced losses.

Finally, we believe that the idea of estimating the BER of a received frame from SoftPHY hints has wider implications beyond just bit rate adaptation, and the interface developed in the context of SoftRate can be used by a variety of future cross-layer protocols.

2. RELATED WORK

We begin by noting that bit rate adaptation is a distinct problem from error recovery. In particular, SoftRate operates with a variety of error recovery schemes, including advanced hybrid ARQ techniques that are more efficient than the "retry entire frame" method used in 802.11a/b/g today. In general, the term "Hybrid ARQ" refers to any scheme that combines forward error correction (FEC) and automatic repeat request (ARQ). Systems such as WiMax [11], cellular high-speed downlink packet access (HSDPA), and more recent proposals such as ZipTx [15] use a form of hybrid ARQ called *incremental redundancy* [16, 17] to match coding rate to channel capacity. Incremental redundancy forgoes aggressive FEC on the first transmission of a packet, requesting subsequent transmissions of parity bits with ARQ only if needed. Partial packet recovery (PPR) [12] is another error recovery scheme that uses SoftPHY hints to retransmit (mostly) only those bits believed to be in error. While these error recovery schemes improve capacity in a time-varying wireless channel, their performance is still contingent on choosing appropriate bit rates for individual transmissions. In other words, while error recovery chooses *which data* to transmit, rate adaptation chooses at *which bit rate* to transmit.

The rest of this section summarizes previous frame-level (§2.1) and SNR-based (§2.2) bit rate adaptation protocols.

2.1 Frame-level Bit Rate Adaptation

Many frame-level rate adaptation schemes have been proposed [14, 18], the most recent ones being RRAA [24] and SampleRate [4], which also provide a good survey of frame-level schemes in general. Frame-level schemes are, by design, less responsive to channel variations than SoftRate because one requires multiple frame receptions to accurately estimate channel state at any bit rate.

SampleRate is currently used in the Linux 802.11 device driver for Atheros cards. It picks the bit rate that minimizes the ten-second average packet transmission time (including MAC layer delays), periodically sampling from bit rates other than the current best in order to adapt to changing channel conditions. RRAA uses short-term frame loss information gathered over tens of frames to adapt bit rate more opportunistically than SampleRate. RRAA also compares the frame loss statistics both with and without RTS/CTS in order to guess whether each loss is caused by a collision or fading on the channel. It then adaptively enables RTS/CTS more frequently as collision losses increase. We compare SoftRate to both (§6), showing significant performance improvements.

COLLIE [20] makes the observation that collision losses adversely impact the performance of rate adaptation protocols. To address this problem, a COLLIE sender analyzes the patterns of bit errors in receptions in order to infer whether an error was due to a collision or a channel loss, and modifies rate adaptation protocols to adapt bit rate on channel losses alone. However, to detect bit errors, the COLLIE receiver echoes the entire received frame to the sender, incurring significant overhead.

Finally, other protocols, [1, 3] use timing information from the physical layer (such as "channel busy" time from Madwifi or packet interarrival times) to infer interference losses, but are susceptible to the same inefficiencies as frame-level protocols in general.

2.2 SNR-based Rate Adaptation Protocols

Because the theoretical relationship between SNR and channel BER is well-known across the various bit rates, it is conceivable that SNR estimates of received frames can be used to pick the best transmit bit rate that maximizes throughput. RBAR [10] uses the RTS/CTS exchange at the beginning of a packet to estimate SNR at the receiver, and picks the transmit bit rate accordingly. OAR

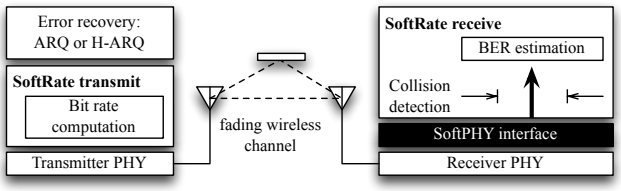


Figure 2: A high-level view of the SoftRate system.

[21] builds on RBAR, opportunistically transmitting back-to-back frames when the channel quality is good. CHARM [13] leverages reciprocity of the wireless channel to estimate average SNR at the receiver using packets overheard from the receiver, thereby avoiding the overhead of RTS/CTS and enabling implementation on commodity cards.

However, it is difficult to accurately measure SNR in current commodity 802.11 systems due to hardware calibration issues and interfering transmissions, as Zhang et al. note [25]. Worse still, the SNR-BER relationship changes with different propagation environments, because the SNR measured at the start of the packet (e.g., using the Schmidl-Cox algorithm [22]) does not capture the variation in SNR that might occur during the packet transmission due to fading. In more recent work, Camp and Knightly [5] evaluate a number of SNR-based rate adaptation protocols and find that because the SNR-BER relationships change with varying degrees of mobility, SNR-based protocols require in-situ training to perform efficiently across different propagation environments. While CHARM [13] proposes a mechanism to calibrate its algorithm on a slower timescale to handle heterogeneous hardware, their mechanism is not effective against changes in the propagation environment that occur on a faster timescale, say, due to more mobility.

Other communication systems like IS-856 CDMA cellular data (1x EVDO) that perform rate-adaptation based on SNR use *pilots* to track the average SNR over the entire duration of the packet, instead of just at the beginning of a transmission, thereby avoiding the problem of sensitivity to propagation environment discussed above. However, in addition to incurring the overhead of pilots, these techniques also cannot differentiate between reduction in SNR due to channel fading and interference. Still, SNR-based bit rate adaptation works well in these systems because they do not experience significant interference from other transmitters by design; these techniques may fail in wireless LANs that experience non-trivial co-channel interference.

We conclude this section with the observation that SoftRate, in spite of being a physical-layer metric-based scheme itself, does not suffer from the pitfalls of SNR-based schemes. Because SoftPHY hints directly estimate packet BER, they do not require any environment-specific or hardware-specific calibration. Moreover, SoftPHY hints along with an interference-detection heuristic can track the variation in the interference-free channel BER across the entire packet without the additional overhead of pilots.

3. DESIGN

This section presents the design of SoftRate. We start by discussing our design goals and giving an overview of the system.

Like other link-layer bit rate adaptation protocols, SoftRate aims to maximize throughput. The link-layer throughput achieved at a certain channel BER and bit rate depends on the error recovery mechanism used (e.g., does the link layer retransmit entire frames, or only the bits in error?). Therefore, SoftRate's use of BER as the bit rate adaptation signal has two benefits:

1. *BER is an accurate predictor of performance.* It is a sufficient statistic that predicts the throughput of various error recovery protocols; as a result, SoftRate cleanly integrates with many error recovery schemes, as we show later.
2. *BER is responsive.* It can be calculated over short timescales on the order of individual frame transmissions, which allows SoftRate to respond to rapid changes in channel conditions.

The SoftRate protocol works as follows. The SoftRate receiver uses SoftPHY hints exported by the PHY to compute the average BER for each received frame (§3.1), employing a heuristic to detect and excise those portions of the frame subject to strong interference (§3.2). The SoftRate receiver then sends the interference-free BER estimate to the sender in a link-layer feedback frame. At the sender's link layer, the SoftRate algorithm (§3.3) uses the per-frame BER feedback to pick the best transmit bit rate for the next frame.

To ensure reliable delivery of feedback, SoftRate always sends its link-layer feedback frame at the lowest available bit rate in a "reserved" time slot, much like 802.11 link-layer ACKs. Feedback is sent whether or not the frame was in error, as long as the frame's preamble and header were decoded correctly. To correctly determine the identities of the sender and receiver even when the frame has an error, link-layer headers are protected with a separate CRC. If the frame has no errors, then the BER feedback is one component of the link-layer ACK. Thus the SoftRate protocol incurs very little extra overhead compared to existing protocols—a CRC in the link-layer header, and a BER measurement in the link-layer ACK. If the sender does not receive any feedback for a frame, the most likely cause is a noisy channel preventing the receiver from even detecting the frame. Therefore SoftRate moves to a lower bit rate if it does not receive feedback for a few consecutive frames (§3.2).

Figure 2 shows how SoftRate fits in the layered network architecture. SoftRate operates using only information provided via the layered SoftPHY interface and can inter-operate with any PHY that is capable of estimating bit-confidences. Our particular design works for any PHY that uses a linear convolutional or block code, which essentially covers all practical wireless systems of interest.

3.1 Estimating BER with SoftPHY

We first show how to compute SoftPHY hints for any PHY using a linear convolutional or block code, examples of which include WiFi, WiMax, and Zigbee. Our approach uses extra information that can be easily obtained and exported from existing decoders.

Suppose x_k, $k = 1 \ldots N$ are the input bits to the encoder at the sender. At the receiver, let $\mathbf{r}$ denote the received signal input to the corresponding decoder. We propose the use of a *maximum likelihood* (ML) or a *maximum aposteriori probability* (MAP) decoder with soft outputs (e.g., Viterbi [6] with soft outputs [8], or BCJR [2]) at the receiver. The output of such a decoder is not bits, but rather *log likelihood ratios* (LLRs) for each received bit, where

$$\text{LLR}(k) = \log \frac{P(x_k = 1|\mathbf{r})}{P(x_k = 0|\mathbf{r})}. \quad (1)$$

Let y_k, $k = 1 \ldots N$ denote the output from the decoder at the receiver. Given the LLRs, the receiver simply "slices" LLR(k) to determine the decoded output bit y_k:

$$y_k = \begin{cases} 1 & : \quad \text{LLR}(k) \geq 0 \\ 0 & : \quad \text{LLR}(k) < 0 \end{cases}. \quad (2)$$

We define s_k as the SoftPHY hint for bit k, where $s_k = |LLR(k)|$.

From SoftPHY to BER. Define the probability of bit error as $p_k = P(x_k \neq y_k|\mathbf{r})$. Then the SoftPHY hint s_k and p_k are related as follows:

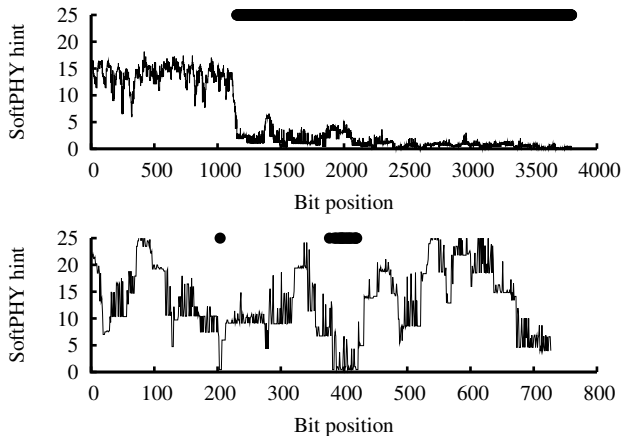


Figure 3: Patterns of SoftPHY hints for a frame lost due to a collision (upper) and due to channel fading (lower). A circle over a bit-position at the top of the graph indicates a bit error.

$$
\begin{aligned}
s_k &= |\mathrm{LLR}(k)| \\
&= \begin{cases} \log \frac{P(x_k=1|\mathbf{r})}{P(x_k=0|\mathbf{r})} & : \quad y_k = 1 \\ \log \frac{P(x_k=0|\mathbf{r})}{P(x_k=1|\mathbf{r})} & : \quad y_k = 0 \end{cases} \\
&= \log \frac{P(x_k = y_k|\mathbf{r})}{P(x_k \neq y_k|\mathbf{r})} = \log \frac{1-p_k}{p_k}.
\end{aligned}
$$

Solving for p_k,

$$p_k = \frac{1}{1+e^{s_k}}. \tag{3}$$

The average of p_k over all bits k in a frame thus gives us the average BER of the channel during the frame transmission.

3.2 Interference Detection

A bit rate adaptation algorithm that reduces the transmit bit rate in response to interference losses increases the contention on the channel and exacerbates the interference. However, a responsive rate adaptation algorithm that reacts to short-term frame loss rate or BER faces the danger of reacting aggressively to interference. To avoid this problem, SoftRate uses a heuristic to identify and separate bit errors caused by interference, thereby adapting the bit rate only in response to the *interference-free BER* of a frame.

If the interferer's signal starts after the receiver synchronizes with the sender's frame, then the interference will manifest itself as a sudden spike in the BER estimated from SoftPHY hints[1]. A sudden change in BER by orders of magnitude within a small number of bits cannot be explained by stochastic channel fading, whose physics are more gradual. For example, Figure 3 contrasts the patterns of SoftPHY hints for a frame that was in error due to a collision and a frame that had bit errors resulting from fading in a mobile channel. A SoftRate receiver uses this heuristic to test every received frame for the presence of interference, and computes the BER of the frame over the interference-free portions alone.

On the other hand, if the sender's signal starts after the receiver synchronizes with the interferer's frame, then the receiver will neither detect the sender's frame nor send the BER feedback to the sender. We call such losses *silent losses*. The sender, on not receiving any feedback from the receiver, does not know if the loss is due to weak signal at the receiver that prevented the sender's frame from even being detected or due to a collision. To avoid this confusion, we propose adding a "postamble" to every frame [12], which enables the receiver to detect with high probability the portion of the sender's frame that lasts after the interference has ended. When postambles are used, the SoftRate sender can assume that consecutive silent losses indicate weak signal at the receiver and reduce the transmit bit rate.

[1]If the interferer's signal is much stronger than the sender's, some PHYs will resynchronize with the interferer and *abort* the sender's frame, which can also be used as a sign of interference.

Frame size of s_1	**Frame size of s_2**	f_1	f_2
1400 bytes	1400 bytes	12%	12%
100 bytes	1400 bytes	14%	1%

Table 1: Fraction of frames f_1 and f_2 at the two senders s_1 and s_2 for which both the preamble and postamble are lost.

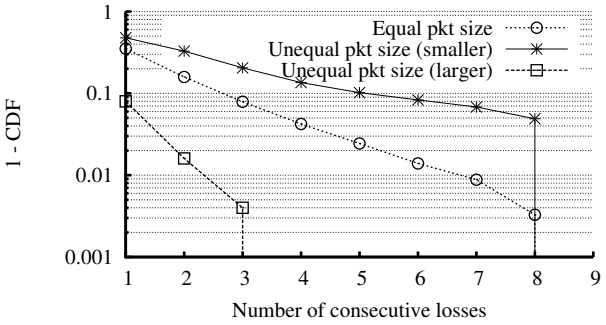


Figure 4: Complementary CDF of run length of consecutive frames whose preamble and postamble are undetected at the corresponding receiver.

A perceptive reader may note that in the cases when the frame durations of the interferer and sender are different (as is likely in multi-rate settings), the sender's frame may fully overlap with the interferer's, resulting in a loss of both the preamble and the postamble. However, we observe that such a situation is unlikely to repeat on a retry of both the frames, because channel access protocols typically implement a backoff mechanism on a frame loss, which changes the relative alignment between the frames on the retry.

To measure the frequency of silent losses due to interference, we use ns-3 to simulate collisions between two nodes that cannot carrier sense each other. We modify the ns-3 802.11 protocol to append and detect postambles at the end of frames. In our simulation, the two senders s_1 and s_2 transmit UDP packets as fast as possible, picking a random transmit bit rate on each packet. The physical layer parameters of the simulation are set such that only collisions result in frame losses, i.e., there are no noise losses. For each sender s_i, we measure the fraction of frames sent f_i for which neither the preamble nor postamble is interference-free and hence decodable at the corresponding receiver. Table 1 shows the fractions f_i for simulations with different frame sizes of the two senders; we find that this fraction is under 15% always. For this small fraction of frames that did lose both the preamble and postamble, Figure 4 shows the complementary CDF of the run length of consecutive losses at the receivers. We infer from the figure that long runs of losses (say, of length 3 or more) are very uncommon due to interference alone. Therefore, a SoftRate sender assumes that three consecutive silent losses indicate a weak signal at the receiver and lowers the transmit bit rate.

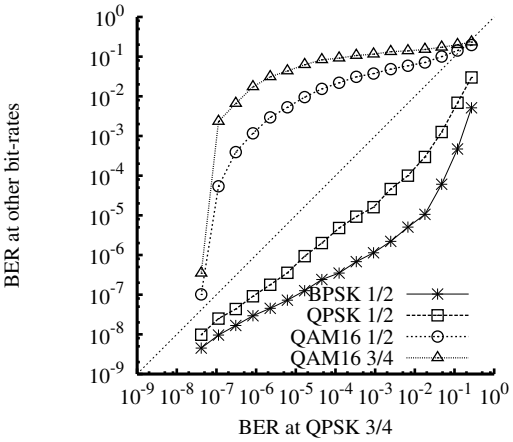


Figure 5: BER at the QPSK 3/4 rate vs. BER at other bit rates from Table 2, using data from the walking trace in Table 4.

3.3 The SoftRate Algorithm

The design of SoftRate centers on three main mechanisms.

1. It uses a heuristic to predict channel BER at a few other bit rates using the BER estimate at one bit rate.
2. Using the above BER prediction heuristic, it computes *optimal thresholds* α_i and β_i for each rate R_i such that, when the BER at rate R_i is in the range (α_i, β_i), then R_i is the optimal transmit bit rate. The computation of the thresholds depends on the error recovery mechanism employed by the link layer.
3. Given the interference-free BER estimate from the receiver and optimal thresholds at each bit rate, the SoftRate sender adjusts its bit rate in the direction of the optimal rate.

Note that SoftRate works when conditions experienced on the upcoming transmission are similar to those on the previous transmission. A wide variety of situations satisfy this criterion (§3.4).

BER prediction. If one knew the detailed relationship between the BER and SNR for each bit rate, then the problem of predicting BER at multiple bit rates using the BER at the current rate would be an easy one. One could simply look up the SNR corresponding to the BER for the current bit rate, and then consult the various SNR-BER curves to determine the BER at each of the other rates. Unfortunately, because the SNR-BER curves depend heavily on both the characteristics of the radio and the environment (see §5.2), this method is unlikely to work robustly.

Instead of relying on SNR-BER relationships, SoftRate uses the two observations below to predict BER.

1. At any SNR, the BER is a monotonically increasing function of the bit rate (this observation is well-known and used by many other protocols).
2. Within the BER range that a bit rate is usable (i.e., BER below 10^{-2}), its BER at a given SNR is at least a factor of 10 higher than that of the next-lower bit rate.

The second observation is general enough to hold in practice, independent of radio and environment characteristics. To see why, note that system designers avoid redundancy in bit rates and offer a set of rates that have at least an order of magnitude difference in error performance at a given SNR. And even if the second observation does not hold in the system, the rate adaptation algorithm can always pick a subset of rates with the above property and use those rates alone for rate adaptation. For example, Figure 5 shows a plot of the measured BER at the QPSK 3/4 rate plotted against the measured BER at two of the higher and two of the lower bit rates; one can see that both the observations above hold on this data. We now show how the approximate BER prediction using the above two observations is sufficient for SoftRate's operation, obviating the difficult problem of estimating the SNR-BER relationships.

Computing optimal thresholds. SoftRate uses the BER prediction method described above to compute optimal thresholds for each bit rate. For each available rate R_i, SoftRate computes α_i and β_i such that, if the BER at R_i is in the range (α_i, β_i) then R_i is the optimal bit rate. Computation of these thresholds depends on the link layer's error recovery mechanism; we will illustrate how to compute thresholds for two such error recovery mechanisms.

Consider the computation of optimal thresholds for the 802.11 a/g 18 Mbps bit rate with frame-level ARQ. If the next lower bit rate is 12 Mbps, then until the BER gets to the point where the frame loss rate is 1/3, the sender should remain at 18 Mbps. For a packet size of 10000 bits, that BER would be of the order 10^{-5}. Now, if the BER at 18 Mbps is lower than, say, 10^{-7}, then the next higher rate of 24 Mbps is likely to have a low enough BER to see no frame losses and hence have a higher throughput. Therefore, the optimal thresholds for the 18 Mbps rate would be $(10^{-7}, 10^{-5})$. In contrast, for some smarter ARQ scheme that can recover from a few bit errors easily by retransmitting a small number of parity bits, the throughput at 18 Mbps may be higher than that at 12 Mbps for up to a much higher BER, say, 10^{-3}. The optimal thresholds for such a link layer would be set to $(10^{-5}, 10^{-3})$.

It is clear that using BER as the information signal helps SoftRate integrate cleanly with many different kinds of error recovery protocols, only requiring a recomputing of thresholds to work with a different error recovery scheme. Frame-level protocols lack this modularity because they consider the frame loss rate in making their decisions, tacitly assuming that entire frame retransmissions are used to recover lost frames. As a result, the bit rate adaptation mechanism itself would have to change if the error recovery protocol changed. Architecturally, our proposal decouples rate adaptation from error recovery and separates the two distinct concerns.

Bit rate selection. Given the optimal thresholds, SoftRate's rate selection algorithm is as follows. Let the current transmit bit rate at a sender be R_i, and let b_i be the most recent interference-free BER estimate at R_i obtained from the receiver. By the design of optimal thresholds, if $b_i < \alpha_i$, then the throughput at the next higher bit rate R_{i+1} will exceed the throughput at rate R_i. Conversely, when $b_i > \beta_i$, the throughput at rate R_{i-1} will exceed that at rate R_i. Therefore, the sender increases bit rate if $b_i < \alpha_i$, lowers bit rate if $b_i > \beta_i$, and does nothing if $b_i \in (\alpha_i, \beta_i)$.

If b_i is far from the range (α_i, β_i), then we can do better by jumping multiple levels to a better bit rate. In the example above, if the BER at 18 Mbps is above 10^{-2}, then one can jump two rates lower to find a bit rate that has a BER under 10^{-5}, as per Figure 5. In general, one can find n levels of rate increase and decrease thresholds α_i^n and β_i^n for every rate i, using which the algorithm jumps n bit rates at a time in the direction of the best bit rate. Our current implementation does up to two rate jumps at a time.

3.4 Behavior Over Time-Varying Channels

We now discuss why the algorithm described above works well across a wide range of wireless propagation environments. There are two sources of variation in the sender's signal at the receiver:

1. Changes in the large-scale attenuation of the signal, often due to changes in distance between the sender and receiver.
2. Multipath fading, the result of multiple copies of a signal being time- or frequency-offset due to mobility.

The coherence time of the channel is approximately the duration of time over which multipath fading effects are expected to stay the

Modulation	Code Rate	802.11 Rate	Implemented?
BPSK	1/2	6 Mbps	Yes
BPSK	3/4	9 Mbps	Yes
QPSK	1/2	12 Mbps	Yes
QPSK	3/4	18 Mbps	Yes
QAM16	1/2	24 Mbps	Yes
QAM16	3/4	36 Mbps	Yes
QAM64	1/2	48 Mbps	No
QAM64	2/3	54 Mbps	No

Table 2: Combinations of modulations and coding rates used in 802.11, the raw throughput achieved over a 20 MHz channel, and their implementation status in our prototype.

same. In a *slow fading* channel, which occurs at walking speeds (or even when the nodes are static, but objects in the environment aren't), coherence times are tens of milliseconds long. Hence, fading and attenuation happen at a timescale corresponding to multiple frame transmissions. In such a channel, the sender's signal fades sharply once every 10-100 milliseconds, typically resulting in a burst of frame losses at higher bit rates. In response to such fading-induced changes, SoftRate lowers the bit rate quickly; it also adapts "upwards" quickly, soon after conditions become better. This adaptation also handles changes in channel attenuation.

Fast fading channels occur at vehicular speeds, where the channel coherence time is between 10 and a few hundred microseconds. This duration is shorter than the transmission time of a frame. However, even in this environment, the BER measured by SoftPHY hints accurately reflects the true BER of the channel, as we show in §5.2. As a result, SoftRate converges to the best transmit bit rate that maximizes throughput, and adapts this best bit rate in response to changes in large-scale attenuation.

Bit rate adaptation is a very hard problem when the coherence time of the channel is in between the two extremes of fast and slow fading, say, equal to two or three frame durations [23]. The way to adapt bit rate in such cases using SoftRate is to increase the packet size to turn it into the fast fading case, or decrease the packet size to turn it into the slow fading case (provided that the packet size is big enough to make this feasible).

4. IMPLEMENTATION

Our 802.11a/g-like physical layer builds on the OFDM (Orthogonal Frequency Division Multiplexing) GNU Radio software-defined radio codebase and the USRP hardware. At the transmitter, incoming data passes through a standard rate-1/2 convolutional encoder, after which it is *punctured* at varying code rates. The punctured bits are then mapped to OFDM subcarriers, using either BPSK, QPSK, QAM16, or QAM64 modulation. The combinations of modulations and coding rates used in 802.11a/g and the corresponding raw 802.11 throughput on a 20 MHz channel are shown in Table 2. The decoding process at the receiver first demodulates the received data, and then decodes it using the soft output BCJR decoder [2], which outputs LLRs that are used to compute the SoftPHY hints (§3.1). Our prototype also computes an SNR estimate for each received frame using the Schmidl-Cox method [22]. Our soft output decoder adds negligible overhead in terms of both receiver complexity and per-packet processing cost.

Interference detector. While mapping coded data onto subcarriers in one OFDM symbol, the transmitter interleaves the data onto non-adjacent (in frequency) OFDM subcarriers. This mitigates bit errors from frequency-selective fading, which causes adjacent subcarriers to fade simultaneously. A collision, however, still causes interference on all subcarriers. We therefore detect collisions as sudden jumps in BER between adjoining OFDM symbols.

Suppose we receive a frame of S OFDM symbols, each symbol containing N_{bps} bits, for a total of $N = N_{bps} \cdot S$ bits, with corresponding SoftPHY hints s_k, $k = 1 \ldots N$. First, we compute p_k, $k = 1 \ldots N$ from the s_k (§3.1). Then, we average p_k, N_{bps} bits at a time, to obtain S average BERs $\overline{p}_j$, one for each symbol j:

$$\overline{p}_j = \frac{1}{N_{\text{bps}}} \sum_{i=1}^{N_{\text{bps}}} p_{i+(j-1)\cdot N_{\text{bps}}}. \quad (4)$$

Finally, our collision detection algorithm is a simple threshold on the difference $d_j = \left|\overline{p}_j - \overline{p}_{j-1}\right|$. Note that this algorithm detects interference that starts after the receiver has synchronized with the signal of interest from a sender. We have not yet implemented the postamble detection logic that will enable identification of interference that starts before the signal of interest.

If the PHY uses time interleaving of bits in a frame, then the bit errors that occur due to interference will be dispersed all over the frame. In such cases, interference detection must be performed before the deinterleaving to capture the temporal patterns of bit confidences. If the deinterleaving occurs before decoding (i.e., before SoftPHY hints are generated), then the interference detection algorithm can work on the inputs to the decoder as well. We note that if the PHY uses some form of interference cancellation [9, 7], then the interference detection strategy remains the same, though the fraction of time interference-related losses occur may be lower.

Fading channel simulator. We implement a Rayleigh fading channel simulator in GNU Radio using a Jakes simulator model [26]. We use the channel simulator to connect the software radio sender and receiver blocks in a local loopback configuration to test our implementation in a variety of channel conditions.

4.1 SoftRate Implementation

The high latency incurred in both procuring RF samples from the USRP front-end and sending link-layer BER feedback makes it impractical to implement and evaluate SoftRate using software radios. We therefore simulate SoftRate and other rate adaptation algorithms in the ns-3 network simulator. However, to keep the simulations realistic and to obtain SoftPHY information on receptions, we replace the ns-3 physical layer models with packet traces collected from our live software radio experiments (§6.1).

We modify the ns-3 802.11 acknowledgment frame structure to include a 32-bit estimate of the received frame's interference-free bit error rate. We also simulate postamble detection; when this option is enabled, the receiver sends an acknowledgment even if the preamble is not detected but the postamble is interference-free.

5. SOFTPHY EVALUATION

SoftRate relies on the following properties of SoftPHY hints:

1. SoftPHY hints can accurately estimate channel BER across a wide variety of wireless propagation channels.
2. SoftPHY hints can be used to distinguish interference losses from fading losses.

In this section, we experimentally evaluate the SoftPHY hints we introduced in §3.1 to verify the above two points, in order to establish their utility in the SoftRate algorithm.

5.1 Method

We present a combination of live experiments and controlled simulations using our OFDM prototype. We subsequently evaluate SoftPHY hints and SoftRate using traces from these experiments.

Experiment	Used in	Method
Static	§5.2	Six static sender-receiver pairs operating in the long range mode were used. Each sender transmitted 100 960-byte packets each at 20 different sender transmit powers and 6 different bit rates.
Walking	§5.2, §6.2	One sender transmitting in short range mode was moved at walking speed away from the receiver in 10 experimental runs of 10 seconds each. A total of 4,000 packets per bit rate were transmitted.
Simulation	§5.2, §6.3	A sender and receiver were connected by our GNU Radio fading channel simulator. The Doppler spread of the channel was varied from 40 Hz to 4 KHz. One hundred packets each were transmitted at 20 different transmit powers of the sender at each of the Doppler spread values.
Static (interference)	§5.3	A sender and interferer transmitted packets simultaneously to a receiver in the long range mode. A random jitter of around one packet-time was added between both transmissions. One hundred packets each were transmitted at five different interferer transmit powers and six different bit rates.
Static (short range)	§6.4	Single static sender transmitted packets in short range mode in 10 experimental runs of 10 seconds each. A total of 4,000 packets were transmitted at each of the bit rates across all the runs.

Table 4: A summary of the experiments used to evaluate SoftPHY (§5) and SoftRate (§6).

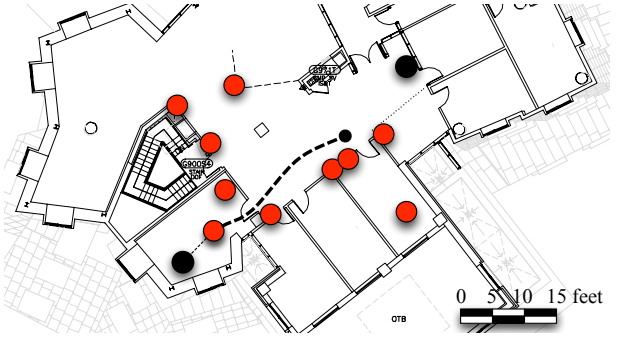


Figure 6: Evaluation testbed: light (red) shaded nodes are senders; black nodes are receivers. The thick dashed line shows the approximate path of the sender in mobility experiments. Background 2.4 GHz traffic provides co-channel interference.

Mode	Bandwidth	Tones	T
Long range	500 KHz	1024	2.6 ms
Short range	4 MHz	512	160 μs
Simulation	20 MHz	128	8 μs

Table 3: Modes of operation of our OFDM prototype. Also shown are the RF bandwidth sampled, number of OFDM subcarriers, and OFDM symbol time T in each mode. The cyclic prefix length is one-fourth the OFDM subcarrier length.

We run our software radio experiments in two modes. In the *long range* mode, the USRP samples a smaller RF bandwidth in the 2.4 GHz band than in the *short range* mode. Because of sampling error, the latter results in signals of lower fidelity from the USRP, resulting in it being unusable over a few links in our testbed. However, a smaller RF bandwidth in the long range mode also leads to typical frame durations of tens of milliseconds. As a result, only experiments in static topologies that see little variation at that timescale were run in the long range mode. In contrast, the short range mode results in frames that last less than a millisecond, making it suitable to run mobility experiments in fading channels that change on shorter timescales. Experiments using our fading channel simulator (instead of the real RF channel) were not limited by the RF front-end; such experiments were run over the normal 20 MHz band with 802.11-like frame durations. We summarize these modes of operation in Table 3. Note that all the modes use more subcarriers than used in commodity 802.11 cards today (i.e., 48) because a higher number of subcarriers enables better physical layer synchronization and channel estimation.

We run a variety of live experiments in static and mobile configurations on the testbed shown in Figure 6. We also run controlled simulation experiments with our fading channel simulator by varying the "Doppler spread" parameter of the fading channel from 40 Hz to 4 KHz. This variation corresponds to channel coherence times[2] between 10 ms and 100 μs, and captures a variety of channel conditions ranging from movement at walking speed in indoor environments (close to 30 ms) to movement at train speeds (close to 100 μs). Table 4 elaborates on the various experiments.

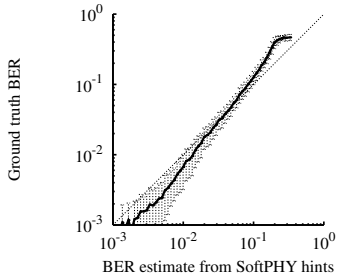
5.2 SoftPHY Reliably Predicts BER

BER prediction in static channels. We first analyze data from the static experiment described in Table 4. For each frame in the trace, we compute the probability of error p_k for each bit k using Equation 3. Then we average p_k over the frame to compute a per-frame average BER. Separately, we determine the frame's ground truth BER by checking the received bits against the known payload. We aggregate the results across different transmit powers, sender-receiver pairs, and bit rates. We bin the BER estimate data in fixed-sized bins of 0.1 units in the SoftPHY metric (roughly logarithmically-sized bins of the estimated BER). Figure 7(a) plots the true BER of the frame against the BER estimated from SoftPHY hints; error bars in this and subsequent figures indicate one standard deviation about the mean. We see from the figure that the SoftPHY-based BER is an excellent predictor of true BER. We also see that the error variance of the SoftPHY BER estimate (across different bit rates) stays below one-tenth of one order of magnitude, implying that it is also a reliable estimator of the true BER.

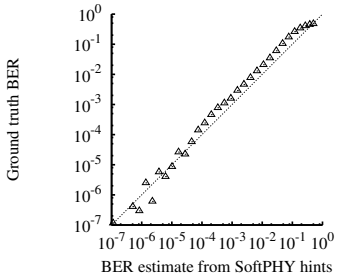
The preceding experiment tests BER prediction frame by frame. But it is hard to reliably observe BERs below 10^{-3} in one 960-byte frame. We therefore aggregate *all* the frames associated with a SoftPHY-based BER prediction bin in the above experiment, and compute the average BER over the aggregated bits (Figure 7(b)). We see that SoftPHY hints accurately predict ground truth BER all the way down to 10^{-7}; the aggregate bits in the graph bins were not sufficient to measure lower BERs.

In contrast, SNR-based BER prediction results in a much less reliable estimate. We separate the trace data by bit rate to analyze SNR-based BER predictions, because the SNR-BER relationship changes for different modulation and coding schemes (unlike in the case of the SoftPHY-BER relationship). Figure 7(c) shows the ground truth BER plotted against the SNR estimate of the frame

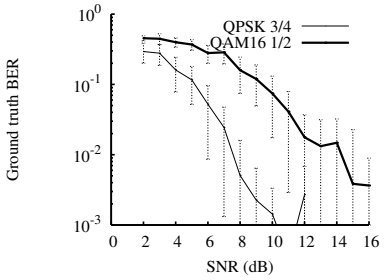
[2]If the Doppler spread in frequency due to mobility is f, then the coherence time of the channel is roughly $\frac{0.4}{f}$ [23].



(a) Per-frame average BER predicted by SoftPHY hints



(b) Aggregated average BER predicted by SoftPHY hints.



(c) Per-frame average BER predicted by SNR.

Figure 7: SoftPHY-based and SNR-based BER estimation in a static wireless channel.

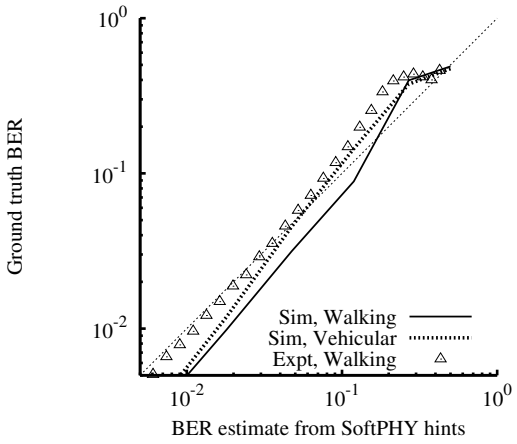


Figure 8: SoftPHY-based BER estimation in a mobile channel.

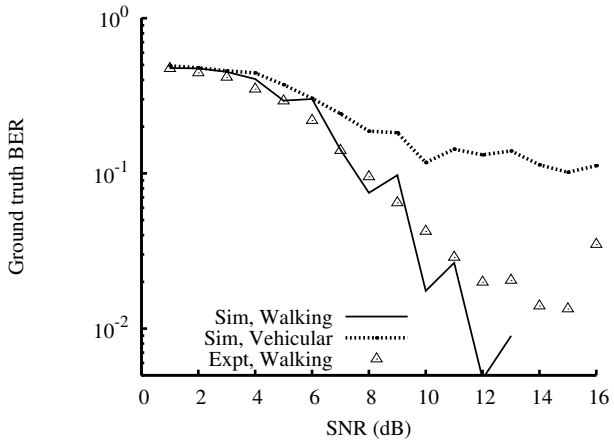


Figure 9: SNR-based BER estimation in a mobile channel.

for two bit rates, with the data binned as described earlier. We observe that a given SNR measurement corresponds to a very wide range of estimated BERs (the estimate has a mean error variance of 2.8×10^{-3} for QPSK 3/4 rate and 1.7×10^{-3} for QAM16 1/2 rate), illustrating that SNR is an unreliable predictor of BER.

BER prediction in mobile channels. We now show that SoftPHY hints reliably estimate BER even in mobile fading channels with widely varying channel coherence times. This section uses data from the walking and simulation traces of Table 4. For each dataset, we bin the data by SoftPHY-estimated BER, and compute the mean ground truth BER in each bin. Figure 8 shows the results, with the two curves corresponding to simulation traces at walking (Doppler spread 40 Hz) and vehicular speeds (Doppler spread 400 Hz), and the points in the figure corresponding to experimental data from the walking traces. Figure 9 shows the corresponding SNR-BER curves at the QAM16 1/2 rate.

From the figures, we see that the SoftPHY-based BER estimate is not sensitive to mobility speed while the SNR-BER curves are. The fact that the SNR-BER relationship changes with channel coherence time is well-known [19] and has also been observed experimentally by Camp and Knightly [5]. This happens because the SNR measured using the preamble does not capture the variation of SNR that happens over the body of the frame in fading channels, which in turn depends on the coherence time of the channel. On the other hand, SoftPHY hints reflect the increasing number of deep fades in the body of the frame as channel coherence time decreases, and therefore estimate BER across all wireless propagation environments accurately. Because the SNR-BER relationship changes with channel coherence time, SNR-based protocols must be carefully retrained for every operating environment; we show later (§6.3) that these protocols pick inaccurate bit rates and suffer a performance penalty if not retrained. In contrast, SoftRate can be used in any wireless propagation environment without retraining.

5.3 Interference Detection Accuracy

We now evaluate our implementation of our SoftPHY-based interference detection algorithm (as described in §3.2 and §4).

False positives. To measure the false positive rate (i.e., the rate at which the fading effects of the wireless channel are falsely identified as collisions), we collect the static and walking traces from Table 4 in a quiet frequency band without any other 802.11a/g transmissions. Out of the resulting frames lost, our collision detection algorithm identified less than 1% of them as collisions.

Interference detection accuracy. We use the traces from the static interference experiment described in Table 4 to measure the accuracy of our interference detection algorithm.

The sender-receiver link in the trace delivered 100% of its frames correctly in the absence of interference. In the presence of interference, one of three things can happen to a frame. First, the frame can be silently lost if the interferer transmits before the sender, ei-

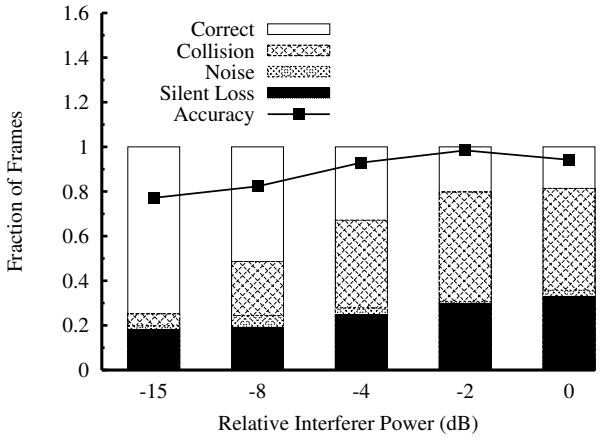


Figure 10: Interference detection accuracy as a function of varying interferer power.

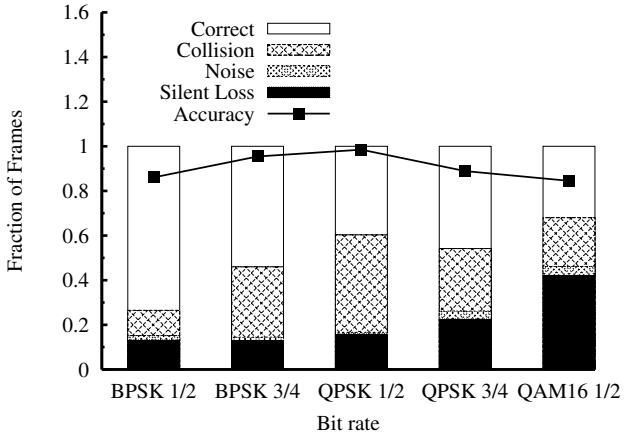


Figure 11: Interference detection accuracy as a function of transmit bit rate.

ther because the receiver has locked on to the interferer's frame, or because the sender's preamble is corrupted by the interferer's signal. Second, the frame can be received, but with errors. Finally, the frame can be correctly received. In the case of frames received with bit errors, we run our interference detection algorithm on the SoftPHY hint traces of the frame to see what fraction of these losses our algorithm identifies as collisions.[3]

We slice the interference detection accuracy results by the different transmit power levels of the interferer and the transmit bit rate of the sender. Figure 10 shows the fraction of frames that fall into each of the cases described above versus the relative interferer strength (in dB). Also shown on the graph is the interference detection accuracy of our algorithm, which is computed as the fraction of frames received with bit errors (i.e., the frames corresponding to "collision" and "noise" in the figure) that our algorithm correctly identifies as collisions. Figure 11 shows the same data, but broken down by the sender's bit rate. We find that our algorithm can always identify more than 80% of frames received in error as collisions. Because the colliding packets are of the same size in this

[3] We omit here results for QAM16 3/4 rate, because our current implementation of that bit rate is untuned.

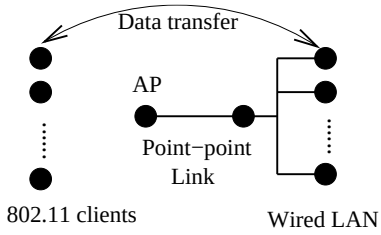


Figure 12: Topology used for the ns-3 evaluation of SoftRate.

experiment, we will be able to detect most of the silent losses as collisions as well by adding postambles.

6. EVALUATION OF SOFTRATE

In this section, we evaluate SoftRate using trace-driven simulations on ns-3, as described in §4. We quantify the performance gains for end-to-end TCP transfers when running SoftRate at the link layer in the following wireless environments: (1) Slow fading mobile channels, (2) Simulated fast fading channels, and (3) Interference-dominated channels.

We use TCP throughput as the metric to evaluate SoftRate against other rate adaptation protocols because applications like TCP and VOIP are more sensitive to losses, and therefore require responsive and accurate rate adaptation protocols to function well. While previous work mostly uses UDP throughput as a measure of performance, we believe that gains obtained on UDP transfers without congestion control are hard to realize in most practical applications.

6.1 Method

Trace-driven simulation. To conduct realistic simulations, we evaluate SoftRate using traces from software radio experiments described in Table 4. For each wireless link being simulated, we seed the simulator with a set of traces, one per bit rate, that completely specify the channel characteristics of the link (like, whether a frame sent is correctly received, and what its SNR and SoftPHY hints would be) for each point in time during the simulation. When the PHY in the simulator receives a frame at a certain bit rate, the fate of the frame is decided by looking up the appropriate trace. The bit rate adaptation protocol at the MAC layer receives and reacts to the feedback from the PHY (frame reception events, SNR estimates, or SoftPHY hints, as the case may be) and sets a suitable bit rate for the next frame. We make no assumptions on the symmetry of links, and use different traces for each of the two uni-directional links between every sender and receiver.

While collecting traces to be used in simulations, we ensure that the channel conditions are consistent across the various bit rates at any point of time. For traces collected using the channel simulator, we simulate the same fading process across experiments at different bit rates. We run live experiments in the short range mode with small frames sent at each of the bit rates in a round robin manner, running through all the bit rates once in under 5 milliseconds. We find that the BER across the various bit rates is monotonic in 96% of such 5 ms cycles, indicating that the channel is indeed fairly invariant across all the bit rates in a 5 ms snapshot.

All traces are collected with one sender transmitting at a time. In simulations with more than one sender, these traces collected in isolation accurately model frame receptions when there are no concurrent transmissions. In case more than two senders transmit simultaneously (e.g., experiments in interference-dominated channels in §6.4), we assume both colliding frames are lost.

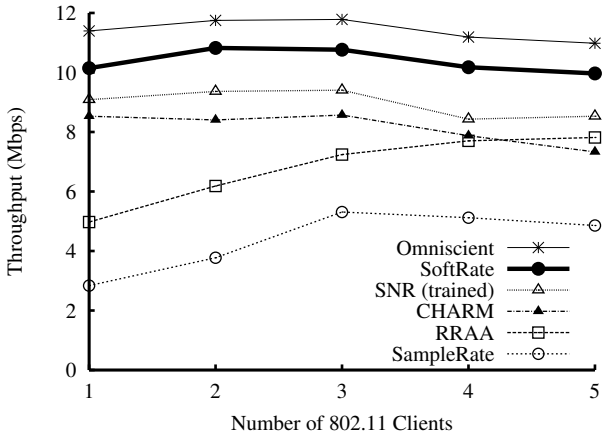


Figure 13: Aggregate TCP throughput (slow-fading mobility).

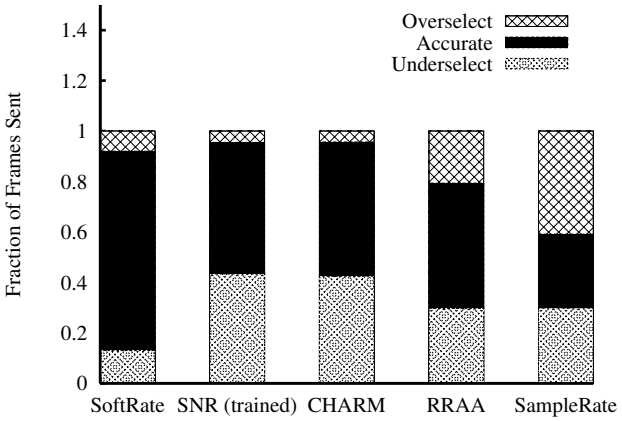


Figure 14: Rate selection accuracy with one TCP flow in a mobile slow fading channel.

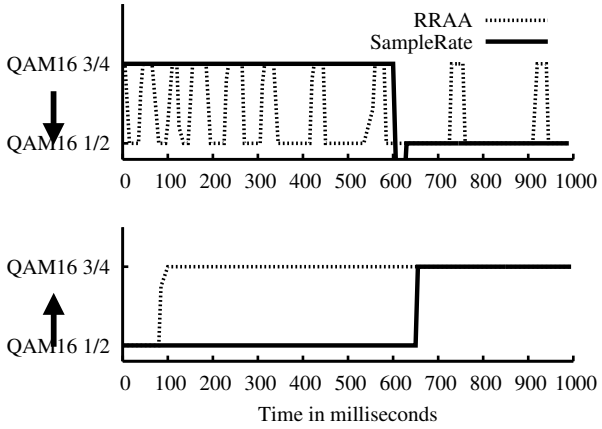


Figure 15: Bit rates chosen by RRAA and SampleRate where the optimal bit rate changes at $t = 0$: from a higher rate to a lower rate (top) and from lower to higher (bottom).

Simulation topology. The topology used in our simulations is shown in Figure 12. N clients connect to an access point (AP) that supports the 802.11a/g bit rates from 6 Mbps to 36 Mbps. The AP is connected to a LAN gateway node by a point-to-point link of bandwidth 50 Mbps and one-way delay of 10 ms. In each experiment, N TCP flows are set up to transfer 1400 byte data frames in either direction between the 802.11 clients and the corresponding wired LAN nodes. Each node's MAC queue length slightly exceeds the bandwidth-delay product of the bottleneck wireless link.

Algorithms evaluated. We compare the performance of SoftRate against the following rate adaptation algorithms.

1. Two SNR-based protocols: (i) a protocol that uses SNR feedback sent via the link-layer ACK to pick the transmit bit rate, much like RBAR but without the RTS/CTS overhead, and (ii) a protocol that uses the average SNR over multiple frames, much like CHARM[4]. The SNR-BER relationships for both protocols are computed from the traces used for evaluation.
2. Two frame-level schemes: (i) RRAA, and (ii) SampleRate. The various parameters in these protocols are set as described in the corresponding references, except for the interval over which transmission time averages are computed in SampleRate, for which a value of one second gave a better performance than the ten second value suggested in [4].
3. An "omniscient" algorithm that always picks the highest rate guaranteed to succeed, which a simulator with a priori knowledge of channel characteristics computes from the traces.

6.2 Slow Fading Mobile Channels

In this section, we evaluate how well SoftRate can adapt to channel variations that occur at walking speeds in a slow fading channel.

Simulation setup. We simulate $N = 1, \ldots 5$ TCP flows from the 802.11 clients to the corresponding wired LAN nodes. We use the ten walking traces (Table 4) to model the ten uni-directional links. We assume perfect carrier sense among all senders.

Results. Figure 13 shows the aggregate TCP throughput obtained by the various rate adaptation algorithms as a function of the number of flows. We find that SoftRate outperforms all other algorithms, and comes closest to the omniscient algorithm. SoftRate gets up to 20% higher throughput than both SNR-based algorithms trained over the traces because the BER prediction from SNR is noisier than that using SoftPHY hints. We also found that using averaged SNR information in CHARM leads to lower responsiveness to short-term SNR variations and hence slightly worse performance than using just the instantaneous SNR value. SoftRate achieves up to 2× higher throughput than RRAA and almost 4× higher throughput than SampleRate because frame-level algorithms cannot adapt fast enough to channel fades that are caused due to mobility, with the result that TCP ends up loosing multiple packets in a window and reduces its offered load. We find that the loss rate experienced by TCP is an order of magnitude higher with frame-level algorithms than it is with SoftRate. We repeat with clients receiving TCP traffic; results are similar to those described above.

For the simulation with one TCP flow, Figure 14 shows how the bit rates picked by the various algorithms on every transmitted frame compared against the highest bit rate that would have gotten the frame through at that time. We find that SoftRate chooses the correct bit rate over 80% of the time.

To better understand the performance of frame-level algorithms, we simulate RRAA and SampleRate using a synthetic trace, where the channel alternates between a "good" state (best transmit bit rate

[4]Our simulation does not need to rely on the channel reciprocity assumptions used in [13] because we can afford to change the 802.11 link-layer ACK in the simulator to piggyback SNR information, while CHARM aims to work with existing 802.11 cards.

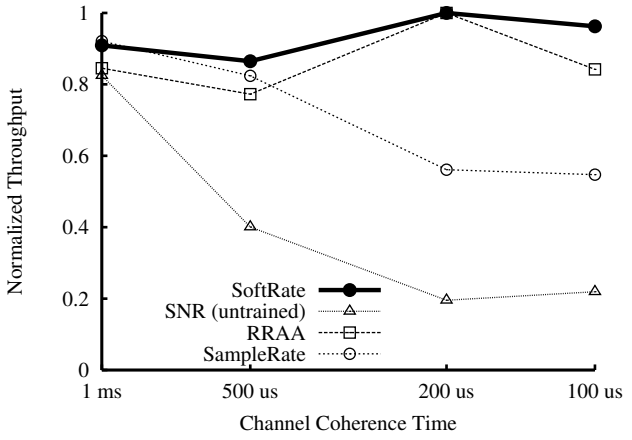


Figure 16: TCP throughput in a simulated fast fading channel.

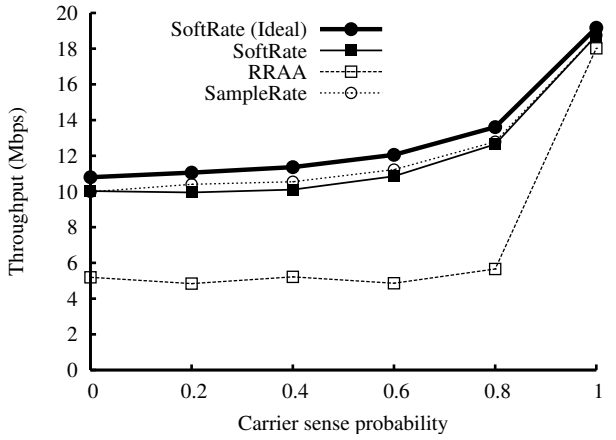


Figure 17: Aggregate TCP throughput as a function of carrier sense probability between the senders.

is QAM16 3/4) and a "bad" state (best transmit bit rate is QAM16 1/2) every 1 second. Frame trace data for the good and bad states are taken from appropriate snapshots in the walking trace described in Table 4. Figure 15 shows the bit rates picked by RRAA and SampleRate as a function of time, where the best transmit bit rate moves from the higher rate to the lower rate in the top panel, and back to higher rate in the bottom panel. The convergence times of RRAA and SampleRate are 15 ms and 600 ms respectively in the first case, and 85 ms and 650 ms in the second. These convergence times explain why the frame-level algorithms frequently overselect and underselect compared to the optimal in Figure 14. One other interesting point to note is the instability of RRAA's rate choice (see the top panel of Figure 15), highlighting another short-coming of frame-level algorithms. When the frame loss rate at a bit rate is zero, frame-level algorithms have no way of knowing if the frames are barely making it through (i.e., the next rate will not work) or if they are getting through very comfortably (i.e., next rate may work). SoftRate knows what the BER at the current rate is and hence can predict whether the next rate will work or not, obviating the need to unnecessarily probe higher rates.

Implications. Failing to adapt the transmit bit rate quickly to channel fades that occur with mobility can lead to burst losses that reduce TCP throughput. As a result, a responsive bit rate adaptation protocol like SoftRate offers huge gains for TCP in mobile channels, compared to less responsive frame-level algorithms.

6.3 Simulated Fast Fading Channels

In this section, we evaluate the performance of SoftRate in fast fading channels that occur at vehicular mobility speeds.

Simulation setup. One 802.11 client transfers TCP data to a wired LAN node via the AP. We use the simulation traces from Table 4 to model the links.

Results. We present the throughput of the various protocols normalized by the throughput of the omniscient algorithm because the best transmit bit rate (and hence the absolute throughput achieved) decreases with channel coherence time. Figure 16 shows the normalized throughput of the TCP flow with various rate adaptation algorithms as a function of varying channel coherence time. The SNR-BER relationships used by the SNR-based protocol are obtained over the walking traces used in §6.2. As channel coherence time reduces, the channel BER at any given bit rate increases for the same SNR. As a result, the SNR-based protocol underestimates the frame BER at lower coherence times and ends up selecting bit rates

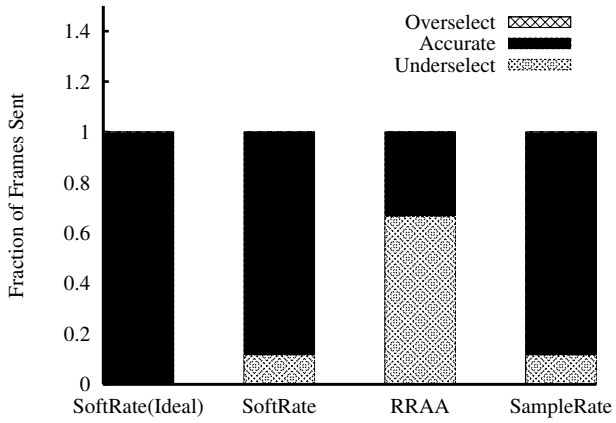


Figure 18: Rate selection accuracy ($\Pr$ [carrier sense**] $= 0.8$).**

that are above optimal. Because SoftPHY hints measure the average BER over the entire frame, SoftRate correctly picks the bit rate that codes for the average channel BER in fast fading channels and its performance stays the same across various coherence times even without retraining. We see from the figure that SoftRate achieves a performance gain of about $4\times$ over the SNR-based protocol at a channel coherence time of 100 μs. Gains over CHARM were similar, as we did not use the retraining mechanism that adjusts SNR thresholds every few seconds, in order to isolate the impact of training on the performance of SNR-based protocols.

Implications. SNR-based protocols incur a performance penalty if not retrained for each operating environment, unlike SoftRate that works robustly across a wide variety of channel conditions without requiring retraining. CHARM proposes to retrain the SNR thresholds on a coarse timescale. However, such mechanisms are meant to handle calibration issues across different hardware and are ineffective if the coherence time of the channel changes on a short timescale, for example, when a train passes by a stationary user.

6.4 Interference-Dominated channels

In this section, we evaluate the impact of interference losses on the performance of SoftRate.

Simulation setup. The simulation consists of five 802.11 clients uploading TCP data via the AP to the wired LAN nodes. We use the static short range traces described in Table 4 to model each of the uni-directional links; using a static channel helps us isolate the benefits due to interference detection from those due to better adaptation in mobile channels. We simulate imperfect carrier sense between the various senders in the simulation to generate collisions. We vary the carrier sense probability between the senders from 0 (i.e., all senders are perfect hidden terminals) to 1 (i.e., perfect carrier sense and hence no interference losses). We simulate two versions of SoftRate–a present version where interference detection succeeds 80% of the time and there is no postamble detection, and a yet-to-be-implemented "ideal" version with postambles and perfect interference detection. When the SoftRate receiver identifies a frame loss as interference, the feedback BER from the receiver is simply the interference-free BER measured in the trace. Otherwise, the feedback is a very high BER indicating a noise loss.

Results. Figure 17 shows the performance of the various algorithms as a function of carrier sense probability. RRAA, because it reacts to short-term frame loss rate, reduces its bit rate in response to interference and sees a much lower throughput than the other algorithms. We found RRAA's Adaptive RTS/CTS scheme to be ineffective in preventing collisions, because interference was unpredictable and resulted in RTS/CTS being constantly turned on and off without any real benefits. SampleRate, on the other hand, is resilient to interference losses because it computes the average transmission time at each bit rate over slower timescales; interference affects the transmission time at all bit rates uniformly at such timescales. The performance of the omniscient algorithm is very similar to that of the ideal SoftRate and is not shown. The performance of the SNR-based algorithms is not affected by interference because the SNR was estimated using the preamble and not over the entire frame. Figure 18 shows the rates picked by the various algorithms on every transmitted frame, compared against the optimal bit rate choice. As expected, RRAA frequently underselects.

Implications. Algorithms that react to short-term channel variations entail the danger of lowering bit rate on interference losses. SoftRate's interference detection mechanism avoids this penalty.

7. CONCLUSION

We have presented SoftRate, a cross-layer wireless bit rate adaptation algorithm that achieves throughput gains of up to $2\times$ over frame-based protocols such as SampleRate and RRAA, 20% over SNR-based protocols trained on the operating environment, and $4\times$ over untrained SNR-based protocols. The key idea is to expose per-bit confidences called SoftPHY hints from the physical layer, using them to estimate the interference-free BER of received frames. Picking bit rates using the BER thus estimated enables SoftRate to react quickly to channel variation without requiring any environment-specific calibration. Moreover, SoftRate's idea of estimating BER from SoftPHY hints can be applied to a variety of wireless cross-layer protocols that, for example, allocate frequency or transmit power, or perform efficient error recovery.

Acknowledgments

We gratefully acknowledge the feedback we received from Brad Karp, our shepherd Ranveer Chandra, and anonymous reviewers. This work was supported in part by National Science Foundation grants CNS-0721702 and CNS-0520032 and by Foxconn.

8. REFERENCES

[1] P. A. K. Acharya, A. Sharma, E. M. Belding, K. C. Almeroth, and D. Papagiannaki. Congestion-Aware Rate Adaptation in Wireless Networks: A Measurement-Driven Approach. In *Proc. IEEE SECON Conf.*, pp. 1–9, San Francisco, CA, June 2008.

[2] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv. Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate (Corresp.). *IEEE Trans. on Information Theory*, 20(2):284–287, 1974.

[3] S. Biaz and N. H. Vaidya. Discriminating Congestion Losses from Wireless Losses Using Inter-arrival Times at the Receiver. In *Proc. of the IEEE ASSET Symp.*, pp. 10–17, Richardson, TX, Mar. 1999.

[4] J. Bicket. Bit-Rate Selection in Wireless Networks. Master's thesis, Massachusetts Institute of Technology, Feb. 2005.

[5] J. Camp and E. Knightly. Modulation Rate Adaptation in Urban and Vehicular Environments: Cross-Layer Implementation and Experimental Evaluation. In *Proc. of the ACM MobiCom Conf.*, pp. 315–326, San Francisco, CA, Sept. 2008.

[6] G. D. Forney, Jr. The Viterbi Algorithm (Invited Paper). *Proc. of the IEEE*, 61(3):268–278, Mar. 1973.

[7] S. Gollakota and D. Katabi. Zigzag Decoding: Combating Hidden Terminals in Wireless Networks. In *Proc. of the ACM SIGCOMM Conf.*, pp. 159–170, Seattle, WA, Aug. 2008.

[8] J. Hagenauer and P. Hoeher. A Viterbi Algorithm with Soft-Decision Outputs and its Applications. In *Proc. IEEE GLOBECOM*, pp. 1680–1686, Dallas, TX, Nov. 1989.

[9] D. Halperin, T. Anderson, and D. Wetherall. Taking the Sting out of Carrier Sense: Interference Canncelation for Wireless LANs. In *ACM MobiCom*, pp. 339–350, San Francisco, CA, Sept. 2008.

[10] G. Holland, N. Vaidya, and P. Bahl. A Rate-Adaptive MAC Protocol for Multihop Wireless Networks. In *Proc. of ACM MobiCom Conf.*, pp. 236–251, Rome, Italy, Sept. 2001.

[11] IEEE Standard 802.16e-2005: Air Interface for Fixed and Mobile Broadband Wireless Access Systems, Amendment 2, Feb. 2006. http://standards.ieee.org/getieee802/802.16.html.

[12] K. Jamieson and H. Balakrishnan. PPR: Partial Packet Recovery for Wireless Networks. In *Proc. ACM SIGCOMM*, pp. 409–420, Kyoto, Japan, August 2007.

[13] G. Judd, X. Wang, and P. Steenkiste. Efficient Channel-aware Rate Adaptation in Dynamic Environments. In *Proc. of the ACM MobiSys Conf.*, pp. 118–131, Breckenridge, CO, June 2008.

[14] A. Kamerman and L. Monteban. WaveLAN II: a High-Performance Wireless LAN for the Unlicensed Band. *Bell Labs Technical Journal*, 2(3):118–133, Summer 1997.

[15] K. C. Lin, N. Kushman, and D. Katabi. ZipTx: Exploiting the Gap Between Bit Errors and Packet Loss. In *Proc. of the ACM MobiCom Conf.*, pp. 351–362, San Francisco, CA, Sept. 2008.

[16] D. Mandelbaum. An Adaptive-Feedback Coding Scheme Using Incremental Redundancy (Corresp.). *IEEE Trans. on Information Theory*, 20(3):388–389, May 1974.

[17] J. Metzner. Improvements in Block-Retransmission Schemes. *IEEE Trans. on Communications*, 27(2):524–532, Feb. 1979.

[18] ONOE Rate Control. http://madwifi.org/browser/trunk/ath_rate/onoe.

[19] J. G. Proakis. *Digital Communications, 4th ed.* McGraw-Hill, 2000.

[20] S. Rayanchu, A. Mishra, D. Agrawal, S. Saha, and S. Banerjee. Diagnosing Wireless Packet Losses in 802.11: Separating Collision from Weak Signal. In *Proc. of IEEE INFOCOM Conf.*, pp. 735–743, Phoenix, AZ, Apr. 2008.

[21] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knightly. Opportunistic Media Access for Multirate Ad Hoc Networks. In *Proc. of ACM MobiCom Conf.*, pp. 24–35, Atlanta, GA, Sept. 2002.

[22] T. M. Schmidl and D. C. Cox. Robust Frequency and Timing Synchroniation for OFDM. *IEEE Trans. on Communications.*, 45:1613–1621, Dec. 1997.

[23] D. Tse and P. Viswanath. *Fundamentals of Wireless Communication*. Cambridge Univ. Press, 2005.

[24] S. Wong, H. Yang, S. Lu, and V. Bharghavan. Robust Rate Adaptation for 802.11 Wireless Networks. In *Proc. of ACM MobiCom Conf.*, pp. 146–157, Los Angeles, CA, Sept. 2006.

[25] J. Zhang, K. Tan, J. Zhao, H. Wu, and Y. Zhang. A Practical SNR-Guided Rate Adaptation. In *Proc. of the IEEE INFOCOM Conf.*, pp. 2083–2091, Phoenix, AZ, Apr. 2008.

[26] Y. Zheng and C. Xiao. Simulation Models With Correct Statistical Properties for Rayleigh Fading Channels. *IEEE Trans. on Communications*, 51(6):920–928, 2003.