Anticipatory Wireless Bitrate Control for Blocks

Paper ID: 150

ABSTRACT

In this paper, we present BlockRate, a wireless bitrate adaptation algorithm designed for *blocks*, or large contiguous units of transmitted data, as opposed to small packets. Our work is motivated by the observation that recent research results as well as widely deployed MAC protocols (such as 802.11n) suggest significant overhead amortization benefits of blocks, yet state-of-the-art bitrate adaptation algorithms are optimized for adaptation on a per-packet basis. As a result, state-of-the-art algorithms can either have the amortization benefits of blocks or high responsiveness to underlying channel conditions of packets, but not both.

To bridge this disparity, BlockRate employs multiple bitrates within a block that are predictive of future channel conditions. In each feedback round, BlockRate uses a history-based scheme to predict the SNR for packets within the next block. In slow-changing scenarios as under indoor mobility, BlockRate uses a simple linear regression model to predict the SNR trend over the next block. In fast-changing scenarios as under vehicular mobility, BlockRate uses a path loss model to capture more significant SNR variations within a block. We have implemented a prototype of BlockRate in a commodity 802.11 driver and evaluated it via deployment on an indoor mesh testbed as well as an outdoor vehicular testbed. Our evaluation shows that BlockRate achieves up to 1.4× and 2.8× improvement in goodput under indoor and outdoor mobility respectively.

1. INTRODUCTION

Bitrate control is a critical component of a wireless protocol stack. Wireless bitrate control seeks to control the effective rate of transmission by adapting the amount of redundancy in transmitted data to the underlying channel quality so as to optimize the received goodput. Although model-based bit rate adaptation in response to channel measurements has been widely considered by the PHY community [7, 3], in recent times a variety of bitrate control schemes [12, 9, 19, 18, 20, 13] that employ measurements of the current protocol operating performance have been introduced and experimentally-verified. They are based on metrics of channel quality ranging from packet loss rate [20, 13], packet transmission time [12], signal-to-noise ratio (SNR) [9, 14], symbol-level dispersion [18], bit error rate and PHY-layer hints [19], etc.

Our work is motivated by the growing disparity between state-of-the-art bitrate control algorithms that are optimized to react on a per-packet basis and technology trends that suggest significant performance benefits to amortizing overhead across blocks, or large chunks of contiguous data transmitted as a single unit, consisting of many packets. For example, Li et al [11] demonstrate significant gains in reliable goodput using blocks by reducing the overhead of acknowledgments, timeouts, and backoffs at the link layer and redundant acknowledgments at the transport layer compared to per-packet TCP. Widely deployed commodity 802.11n cards already enable large opportunities of uninterrupted transmission (of up to 64 KB [4]) consisting of many packets. However, state-of-the-art bitrate control algorithms in research as well as in practice continue to be designed with per-packet adaptation in mind. If used as-is with blocks, these algorithms are prone to be unresponsive to changes in underlying channel quality as large blocks imply a commensurately large delay in obtaining feedback about channel quality.

This disparity raises two natural research questions that form the focus of this paper. First, do the performance benefits of large blocks outweigh the performance loss due to the reduced responsiveness of bitrate control to changes in the underlying channel quality? Second, and more importantly, is it possible to have the performance benefits of blocks without compromising on the responsiveness of bitrate control?

Our measurement-based experiments with several static as well as mobile scenarios answer the first question in the affirmative. Our results show that traditional bitrate control algorithms designed to operate on a per-packet basis achieve moderately higher goodput when used as-is with blocks. This net performance benefit shows that there is greater value in amortizing overhead than reacting quickly to channel conditions even in dynamic settings. Furthermore, our results also show that there is room for improvement, i.e., an ideal block-based bitrate control scheme with future knowledge significantly outperforms packet-based bitrate control schemes used as-is with blocks, thereby setting up the stage for the second question.

Our main contribution, the design and implementation of BlockRate, a block-based bitrate control algorithm, affirms the second question as well. The key insight in BlockRate is to use multiple bitrates across packets within a block that are predictive of future channel conditions. While this may sound impossible at first glance, our measurement experiments across several typically encountered pedestrian and vehicular mobility scenarios in indoor as well as outdoor settings show both that it is possible to predict the future SNR and that leveraging this predicted information to select the bitrate translates to significant gains in goodput. BlockRate maintains a receiver-assisted technique to maintain a mapping between the SNR and the best bitrate corresponding to that SNR. BlockRate uses this history-trained SNR-to-bitrate mapping in conjunction with its model for predicting the future SNR to pick (possibly different) bitrates for packets in the next block.

BlockRate uses two simple models to predict the SNR experienced by packets in the near future. The first model is applicable to slow-changing scenarios such as static or pedestrian mobility scenarios. In such slow-changing scenarios, BlockRate employs a *linear regression model* based on a historic time series of SNR values to predict the future SNR. The second model is applicable to fast-changing scenarios as is typical under vehicular mobility. In such fast-changing scenarios, BlockRate employs a *path loss model* in conjunction with the historic time series of SNR values and its knowledge of the distance to the receiver to predict the future SNR.

We implemented a prototype of BlockRate in the MadWiFi driver [2] and deployed it on an indoor mesh testbed and an outdoor vehicular testbed consisting of 35 vehicles moving in and around our city[1]. Our evaluation shows that, compared to existing packet-level schemes used as-is with blocks, BlockRate achieves up to 1.4× and 2.8× improvement in unreliable goodput in pedestrian and vehicular mobility scenarios respectively; and 2.1× improvement in reliable goodput in indoor scenarios.

2. A CASE FOR BLOCK BASED BITRATE CONTROL

In this section, we experimentally motivate the need for block-based bitrate control. Our measurement-based experiments reveal two findings. First, the performance benefits of amortizing overhead with blocks outweigh the performance loss of due to less responsive bitrate control resulting in a net gain. Second, an ideal block-based protocol with future knowledge can significantly improve upon this gain compared to using existing packet-based bitrate control schemes as-is with blocks. We begin with a brief background on the benefit of blocks.

2.1 Background: Blocks versus packets

Although one would expect MAC and transport protocols to be engineered so as to ensure low overhead, that is not the case in practice today. To appreciate this, consider Figure 1 that shows a comparison of the transmission overhead associated with sending a packet and a block in 802.11a/b/g networks. Acquiring a transmission opportunity entails a carrier sensing interval (DIFS) and a possible backoff. A successful receipt incurs a brief holding period (SIFS) and an acknowledgment transmission time. Under good channel conditions, this combined overhead is acceptable. However, poor channel conditions may latch on several rounds of sender timeouts and backoffs of increasing length until a successful transmission. Li et al. [11] show that for reliable transport, cross-layer effects in the form of redundant TCP acknowledgments and negative interactions between TCP rate control and fluctuating link- and network-layer delays further exacerbate the net impact of using small packets as a unit of transmission.

Thus, it is important to amortize overhead by sending as much as possible in each transmission opportunity. Simply sending massive packets is impractical as the likelihood of corruption exponentially increases with the packet size and network-layer MTUs limit the size of packets. Instead, *blocks* consisting of multiple packets have come to be recognized as a better alternative. For example, as illustrated in Figure 1, widely used 802.11n implementations allow for blocks of up to 64 KB [4] in each transmission opportunity. Li et al. [11] advocate blocks of up to 1MB with multiple carrier sensing rounds within each block so as to keep delays for competing interactive connections small.

To illustrate the overhead benefits of using 802.11n-style blocks, consider a back of the envelope calculation based on the values measured by the MadWiFi driver [2]. Assume packets of size 1.5KB and a block comprising of just *three* such packets as shown in Figure 1. The DIFS, SIFS and packet header have transmission times of 28μs, 9μs and 20μs respectively. For packet transmissions, the acknowledgement takes 200μs while the average backoff duration is 436μs; the same values for transmitting a block are 220μs and 528μs[2]. The overhead, i.e., the sum of DIFS, SIFS, header, backoff and acknowledgement, for a bitrate of 12Mbps of transmitting a block is a factor 2.3× lower compared to packets. This reduced overhead in turn translates to a 1.3× improvement in goodput.

[1]Details omitted to preserve anonymity.

[2]These values were obtained based on measurements on a local mesh testbed as detailed in §4

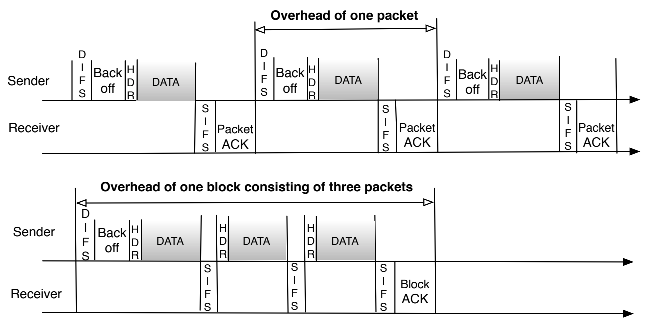


Figure 1: Overhead for packet and block.

2.2 Bitrate control in blocks

Although large blocks help improve performance by amortizing overhead, there is an obvious downside, namely that it severely reduces the responsiveness of bitrate control as feedback about channel quality is delayed by a factor proportional to the size of a block. Commodity 802.11n cards using bitrate control schemes primarily designed for per-packet adaptation attempt to balance this trade-off by simply limiting the block size. The block-based transport protocol, Hop [11], does not address the problem of rate control, experimenting primarily with a fixed hand-picked bitrate. More recent bitrate control schemes optimized for 802.11n [13] leverage the MIMO nature of its PHY layer, but continue to implicitly assume per-packet rate adaptation.

This state-of-the-art leaves open the question of whether existing bitrate control schemes designed with per-packet adaptation in mind are adequate if used as-is with blocks and if not, how much room is there for improvement? We study this question in an indoor static setting and an outdoor vehicular setting in turn next.

2.3 Blocks vs. packets: Static indoor

We conduct a simple experiment on an indoor 802.11g mesh testbed of 16 nodes (Figure 7) to compare the effectiveness of bitrate control using blocks and packets. We randomly pick 30 links and continuously send UDP packets, selecting one link at a time. The packet size is 1.5KB and each block contains 200 packets. A more detailed description of the testbed is deferred to §4.

We compare the performance of two block-based bitrate control algorithms, IdealBlock and CharmBlock, against two packet-based ones, SampleRate [12] and Charm [9], across different links in the testbed. IdealBlock picks for each link the bitrate that is known through recent measurements to be the best bitrate for that link. IdealBlock is intended to serve as a lower bound on the performance achieved by an optimal bitrate control scheme. CharmBlock uses the Charm bitrate control algorithm [9] as-is in conjunction with blocks. Charm and SampleRate are both well-known packet-level bitrate control algorithms that have been shown to work well in 802.11a/b/g networks; the former adapts the bitrate based on the SNR while the latter adapts the bitrate based on the packet transmission time.

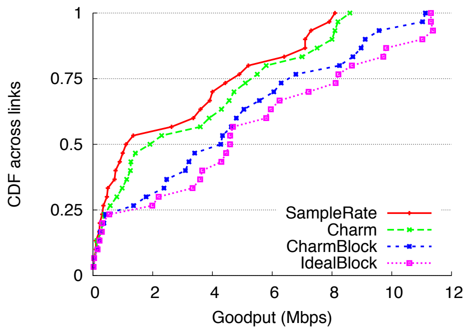


Figure 2: CDF of goodput: Static indoor. IdealBlock achieves a median goodput of 2.1× over Charm.

Figure 2 shows the results for the compared bitrate control algorithms, where each line is the CDF across all links of the goodput achieved by the corresponding algorithm. The experiment yields two important insights. First, the block-based algorithms, IdealBlock and CharmBlock, significantly outperform the corresponding packet-based algorithms. In particular, IdealBlock achieves a median improvement of 2.1× compared to Charm. Second, in static indoor settings, simply using a packet-based algorithm as-is with blocks, as indicated by the CharmBlock line, yields significant gains while leaving some room for improvement. This finding is unsurprising as the goodput gains by amortizing overhead using blocks significantly outweigh the loss due to reduced responsiveness of bitrate control; in an indoor static setting, the channel quality does not change enough to warrant frequent step-ups or step-downs in the bitrate.

Figure 3 shows that the goodput gains result from the bitrates selected by the different schemes. While IdealBlock selects a fixed bitrate for each link, SampleRate and Charm select bitrates for every packet dynamically depending on their assessment of the channel quality. We find that both of the latter schemes select a single *majority* bitrate over 70% of the time, as expected in a static indoor setting. In Figure 3, we show this majority bitrate for each link for these two schemes. The figure shows that IdealBlock selects a higher bitrate than SampleRate and Charm on 16 out of 30 links while CharmBlock does so on 13 occassions. Both SampleRate and Charm happen to select the same bitrate in this experiment.

Why do block-based schemes pick a higher bitrate compared to packet-based schemes? In particular, why does CharmBlock pick a higher bitrate than Charm (over packets)? To appreciate this, recall that all bitrate control algorithms fundamentally seek to maximize goodput, or equivalently $\frac{1-l(r)}{t(r)}$, where $t(r)$ is the average time (including overheads such as carrier sensing, timeouts, acknowledgments, backoffs, etc.) for each

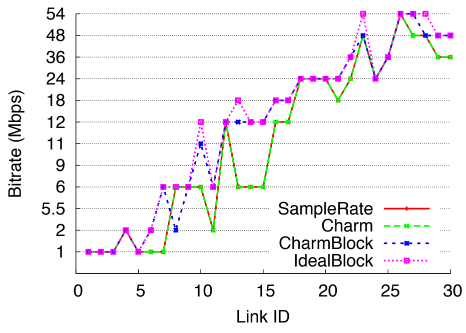


Figure 3: Bitrate selection: Static indoor. IdealBlock selects higher bitrate than SampleRate and Charm on 16 links.

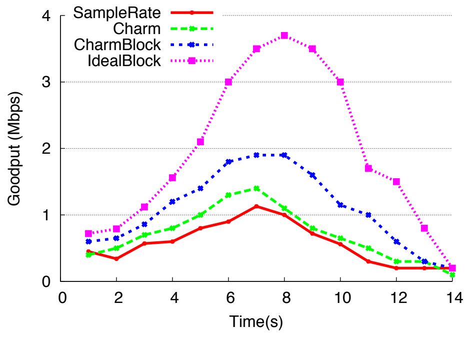


Figure 4: Goodput under vehicular mobility. IdealBlock has up to 2× goodput improvement over CharmBlock.

transmission of a packet and $l(r)$ is the loss rate at the bitrate r. Consider two bitrates r_1 and r_2 ($r_1 < r_2$). For packet-based schemes, suppose the corresponding loss rates are, say, $l(r_1) = 0.1$ and $l(r_2) = 0.3$, and packet transmit times (including overhead) are $t(r_1) = 10$ and $t(r_2) = 9$ respectively.

Packet-based schemes will select the smaller bitrate r_1 as $\frac{1-l(r_1)}{t(r_1)} > \frac{1-l(r_2)}{t(r_2)}$. In comparison, blocks reduce the effective overhead associated with each packet transmission. Suppose the average transmit time of each packet sent as part of a large block is $\tau(r_1) = 8$ and $\tau(r_2) = 5$ at the bitrates r_1 and r_2 respectively. Then, a block-based scheme will select the higher bitrate r_2 as $\frac{1-l(r_1)}{\tau(r_1)} < \frac{1-l(r_2)}{\tau(r_2)}$. Note that in this example, $\tau(r_2)$ decreased by a bigger factor (from 9 to 5) compared to $\tau(r_1)$ (from 10 to 8), which is consistent with the observation that blocks reduce the overhead significantly in poorer conditions, i.e., when loss rates are higher. Thus, blocks enable bitrate control to explore higher (more lossy) bitrates aggressively without fear of a steep increase in loss-induced overhead.

2.4 Blocks vs. packets: Vehicular outdoor

The results above might appear to suggest that existing bitrate control algorithms used as-is with blocks achieve much of the possible performance gains with a small room for improvement. However, the situation is very different in mobile settings with frequent and significant variations in channel quality.

To assess the impact of mobility, we conduct an experiment involving vehicular mobility in an outdoor setting. This experiment involves a laptop placed in a car driven by one of the authors sending back-to-back UDP packets to another stationary laptop acting as an "access point". The packet and block sizes are the same as in the static, indoor experiment described above.

Figure 4 shows the goodput achieved by the compared bitrate control algorithms as a function of time. All schemes show a period of increasing goodput followed by decreasing goodput, which is consistent with the vehicle first moving towards the access point and then away from it. As before, the two block-based schemes, IdealBlock and CharmBlock, significantly outperform packet-based schemes. However, unlike the static indoor setting, we observe a significant difference (of up to 2×) in the goodput achieved by IdealBlock compared to CharmBlock, i.e., a well-designed block-based bitrate control scheme significantly outperforms a packet-based scheme used as-is with blocks.

The rest of the paper describes a practical block-based bitrate control scheme that achieves a goodput comparable to IdealBlock in a variety of mobile scenarios. Above, IdealBlock itself was obtained using a tedious a priori measurement process where one of the authors experimented with all possible bitrates at points spaced 5 meters apart from the access point. The measurements at each point were conducted while remaining stationary; for this reason as well as possible temporal variations in channel quality, we intend for IdealBlock to serve as a lower bound on the ideal goodput possible using blocks.

Summary: The performance gains obtained by amortizing overhead using large blocks more than outweigh the loss due to reduced responsiveness of bitrate control to the underlying channel quality in static as well as in mobile scenarios. However, in mobile scenarios, state-of-the-art bitrate control schemes used as-is with blocks leave significant room for improvement compared to an ideal scheme with future knowledge, making the case for a bitrate control scheme optimized for blocks.

3. BlockRate DESIGN

In this section, we present BlockRate, a block-based bitrate control algorithm that achieves high goodput in variety of static and mobile settings. The key insight in BlockRate is to use multiple bitrates across packets within a block that are selected based on the predicted SNR trend over the transmission of the block.

3.1 Overview

BlockRate uses the SNR measured at the receiver to learn for each SNR regime and bitrate the corresponding packet loss rate. The receiver learns this mapping, referred to as the *SNR-bitrate table*, by monitoring the loss rates of packets sent at various attempted bitrates by the sender in the recent past. The receiver uses this table to select the best bitrate for each packet in the next block based on the predicted SNR for the packet, and conveys this information to the sender piggybacked with a selective acknowledgment for packets within the current block.

The high-level design described above is similar to existing SNR-based bitrate control schemes [9, 14], but with an important difference in the prediction step. Packet-based bitrate control schemes simply assume that channel conditions do not change significantly in the course of a packet or two, so the SNR predicted for the next packet is the same as that of the current packet. However, as shown in the previous section, this simplistic prediction performs poorly at the block granularity.

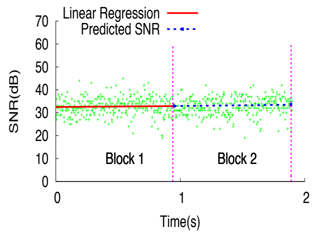
To address this problem, BlockRate uses two simple models to predict the SNR for packets within the next block. The first is a *linear regression model* invoked in slow-changing environments such as in static or pedestrian mobility scenarios. The second is a *path loss model* invoked in fast-changing scenarios such as under vehicular mobility. We explain each of these in detail next.

3.1.1 Linear regression model

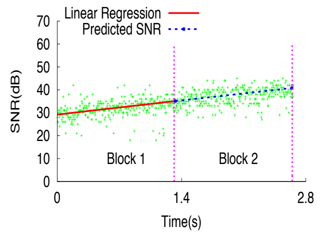
In static or pedestrian mobility scenarios, the SNR trend changes slowly. To appreciate this, consider Figures 5(a) and 5(b) showing the variation in packet SNRs from one block to the other in an indoor setting, for the static and the pedestrian mobility scenarios respectively. To measure the SNR, we use two laptops configured in a sender-receiver mode. In the first experiment, both laptops are kept static (with no line of sight), while in the second one of them is kept stationary with the other being moved towards it at walking speeds.

Figure 5(a) shows that in the static case, the SNR almost remains constant; with most fluctuations confined to a small range of 10 dB. Figure 5(b) shows that in the walking case, the small fluctuations persist, but are accompanied by a weak upward trend as the receiver is moved towards the sender. Depending on the bitrates selected, each block can take between 0.2 and 1.5 seconds to be transmitted. Thus, if a pedestrian user's motion (speed and acceleration) remain unaltered for the next second or two, the SNR trend can be expected to continue across consecutive block transmissions. We find that this is indeed the case throughout our entire set of pedestrian mobility traces (not shown for brevity), not just the specific example shown in Figure 5(b) for illustrative purposes.

In such slow-changing scenarios, a simple linear regression model [1] can effectively capture SNR trends across consecutive blocks. The model assumes that the received SNR has a simple linear relationship with time, i.e., the model fits a straight line through the set of SNR samples using the least squares estimate. BlockRate extends this regression line to predict the SNR variation over the course of the next block.

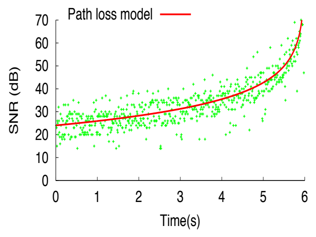


(a) Static

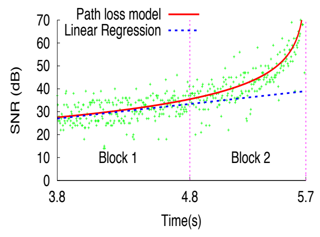


(b) Pedestrian

Figure 5: SNR variations in static/pedestrian mobility



(a) Vehicular



(b) Vehicular (zoomed)

Figure 6: SNR variation: vehicular mobility

3.1.2 Path loss model

In outdoor vehicular mobility scenarios, SNR variations can be rather steep within the course of a block, as the vehicle's distance from the access point can change significantly during that time. For example, Figure 6(a) shows the time series of packet SNRs observed by a receiver kept in a car approaching a sender (AP) at 30 mph. We conducted this experiment in a downtown area using two laptops with the sender laptop placed at a fixed location near the edge of the road. The figure shows that the SNR changes rapidly within several seconds and, more importantly, does not show a simple linear relationship with time (especially in the 3–6 second regime).

Figure 6(b) further zooms into the SNR of packets within two consecutive blocks in this regime. The figure shows that the linear regression line deviates considerably from the actual SNR, so any estimation based on linear extrapolation will likely result in suboptimal performance. More importantly, the SNR shows this sharp, non-linear trend when the vehicle is closest to the AP and has a good channel (SNR $>$ 50 dB). We show in §4.2 that the highest bitrates and goodput are achieved during this time period, so it is critical to capture these sudden changes in SNR. One approach to alleviate this problem might be to reduce the block size so that the linear regression method can continue to be reasonably accurate. However, we find that this significantly impairs the overhead amortization benefits of blocks, thereby hurting goodput.

To address this problem, BlockRate adopts the wireless channel path loss model [15] to model the non-linear variation of SNR over time. The path loss model postulates a logarithmic relationship between path loss and distance:

$$PL(d) = PL(d_0) + 10\alpha \log_{10}(d/d_0) \tag{1}$$

where PL is the path loss in dB, $PL =$ transmit power - receive power, d_0 is the reference distance, and α is the path loss exponent. Since on a dB scale, SNR = transmit power - PL - noise, and as the transmit power at the sender and the noise at the receiver are usually fixed, we can rewrite Eq. 1 as follows:

$$\text{SNR}(d) = \text{SNR}(d_0) - 10\alpha \log_{10}(d/d_0) \tag{2}$$

i.e., SNR increases logarithmically with decreasing distance. Figure 6 shows that the path loss model indeed fits the SNR variations under vehicular mobility well.

To use the path loss model for SNR prediction, BlockRate needs to know the distance d between the sender and the receiver. Our design assumes that GPS devices are readily available in vehicles, and both the sender and the receiver periodically broadcast beacons containing the GPS information. GPS devices can provide a position accuracy of within 3 meters, 95% of the time in an outdoor environment, which we find is sufficient for the model. In this respect, BlockRate is similar to other recent work [16] using external sensor hints from GPS to augment network protocols in different scenarios.

BlockRate assumes that the vehicle moves in a straight line with constant speed between succesive block transmissions, and that the transmission times for two consecutive blocks is the same. The reference distance d_0 is chosen to be 50m and beacon messages are used to estimate d. Consider two successive block transmissions and let d_1 and d_2 be the actual distances (calculated using the GPS beacons) at the beginning of the first and second blocks respectively. At first, the receiver uses these distances and the SNR of received packets to estimate the path loss exponent. Although d changes non-linearly with time we estimate the distance variation to be linear. Therefore at the end of the second block the distance between the sender and receiver is predicted to be $|2d_2 - d_1|$. Knowing this distance variation and the estimated path loss exponent, BlockRate can estimate the SNR variation for the next block using Eq 2. The path loss exponent is updated with every GPS beacon received.

We note that there are many sophisticated prediction algorithms for the wireless channel under various assumptions [6]. It may be feasible to eliminate the need for a positioning system as well as the simplistic assumptions of zero acceleration above. However, this would require more data points and a more sophisticated multivariate regression technique to estimate both the distance as well as the path loss exponent. As our primary goal in this paper is to establish the feasibility of using simple SNR prediction techniques for bitrate control, we defer further refinement of them to future work. Our evaluation shows that, despite its simplistic assumptions, BlockRate performs well in a variety of typically-encountered vehicular mobility scenarios some of which include moderate acceleration.

Bitrate (Mbps)	**1**	$\cdots$	**12**	**18**	**24**	$\cdots$	**54**
Rcvd	0	$\cdots$	72	70	54	$\cdots$	0
Lost	0	$\cdots$	11	19	42	$\cdots$	0

Table 1: SNR-Bitrate table (SNR=45dB).

3.2 Building the SNR-Bitrate mapping table

An SNR-bitrate table is used to select the best bitrate at each SNR. It records the number of received and lost packets at all available bitrates in each SNR regime. Table 1 shows an example of the mapping information at an SNR of 45 dB.

The receiver updates the SNR-bitrate table each time it receives a block. For a correctly received packet in the block, the receiver increments the Rcvd field in the mapping table based on the packet's SNR and bitrate. For a lost packet, the receiver estimates its SNR and bitrate as the SNR and bitrate of the most recent correctly received packet and then increments the Lost field using these values. We let the table keep data obtained in a recent window of five seconds as we did not observe any benefit from having a larger window.

Our estimation of the SNR for the lost packet is based on the assumption that the channel SNR remains unchanged over the course of a small number of packet transmissions. This helps us avoid short-sighted reaction to interference or unpredictable short-term changes in channel quality. We note that similar assumptions have also been made by other SNR-based bitrate control schemes [9].

3.3 Selecting the best bitrate

Given the SNR-bitrate table, BlockRate selects the bitrate that maximizes the goodput at each SNR. The goodput is computed as $b * (1 - l)$, where b is the bitrate and l is the lossrate. In the previous example in Table 1, the bitrates of 12Mbps, 18Mbps and 24Mbps have loss rates of 13%, 21%, 44% respectively, so the corresponding goodputs are 10.4Mbps, 14.2Mbps, and 13.4Mbps respectively. Thus, 18Mbps is the best bitrate for an SNR value of 45 dB.

We observed from our experiments that a continuous range of SNR values tend to be mapped to the same best bitrate in a given environment. Hence, as the packets within a block have gradually changing SNR, the best bitrate for the packets should change following a step pattern, i.e., the best bitrate remains the same for a certain number of packets, jumps to the next higher/lower one, again persists for some packets.

BlockRate uses the above observation to optimize transmission overhead. Once the receiver selects the best bitrate for each packet in the next block based on their predicted SNR, it compacts this information using a runlength-encoded map. We illustrate this using an example. Assume that a block has 200 packets and packets numbered between 1-115 use 12Mbps while those between 116 -200 use 18Mbps. The receiver then constructs a table specifying just the range of packets and the bitrates to be used, in this case [115, 12; 200,18], instead of explicitly specifying this information for every packet. This compact table is then inserted into the block acknowledgement and sent back to the transmitter.

Let us assume the selected set of best bitrates for packets in the next block to be {18Mbps, 24Mbps}. In addition to sending packets at the receiver-recommended bitrates, BlockRate also probes bitrates one level lower and higher in order to detect better transmission opportunities. BlockRate sends 10% of the total packets within a block at each of the two probe bitrates. In the above mentioned example, the sender then randomly picks 10% packets within the block to send at 12Mbps, and another 10% to send at 36Mbps. We observe that a typical block transmission contains between 3 and 4 different bitrates including the probed bitrates in an indoor environment and between 4 and 5 in an outdoor setting. Using this probing mechanism, we are able to exploit enough bitrate diversity and discover better transmission opportunities.

3.4 Implementation

We implemented BlockRate in the MadWiFi driver [2]. Each block consists of 200 packets and each packet is 1.5KB in size. We disable link-layer ARQ to eliminate per-packet acknowledgements. We leverage 802.11e's transmission opportunity (txop) to further reduce channel contention time. A txop is a contention-free burst of link layer packets separated by SIFS. We use the maximum allowed value of 8192μs for txop, which is equivalent to 5 to 8 packet transmissions. A block thus roughly consists of 60-100 txops, and one block transfer takes 0.2 to 1.5 seconds to complete.

The receiver sends a block acknowledgement back to the sender after it receives the whole block. It identifies the end of a block by a "reserved" packet that is always transferred at the lowest bitrate to ensure reliable transfer. We were unable to implement the block acknowledgement completely in the kernel space of the MadWiFi driver, so we designed a user-space module for sending the block acknowledgement. When the acknowledgement is received at the sender, it uses *sysctl* to pass all the required information to the driver so that it can process the new block.

The MadWiFi documentation [2] states that "the reported RSSI for each packet is equivalent to the Signal-to-Noise Ratio (SNR)" and so BlockRate uses the RSSI values reported by the driver as the received SNR, an approach that has also been adopted by other researchers [10]. We obtain outdoor location information using USB GPS devices installed on vehicles in our testbed.

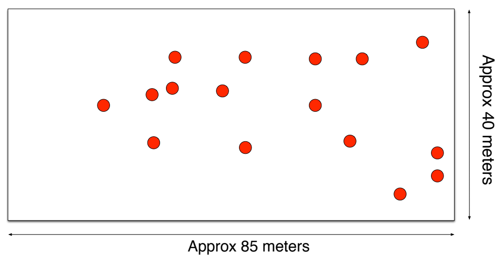


Figure 7: The Mesh testbed showing node locations. The floor plan is replaced with a box for anonymity.

4. EVALUATION

In this section, we compare BlockRate to existing packet-based bitrate adaptation algorithms as well as adaptations of these algorithms for blocks. A summary of our results is as follows.

- ▶ BlockRate improves unreliable goodput over existing algorithms by up to 2.8× under vehicular mobility (§4.2) and by up to 1.4× under pedestrian mobility (§4.3).
- ▶ In conjunction with a block-based reliable transport protocol, BlockRate improves reliable goodput by up to 2.1× in an indoor setting(§4.5).
- ▶ BlockRate is robust to interference effects, i.e., its improvement over packet-based bitrate control algorithms does not change significantly due to interference (§4.4 and §4.5).

4.1 Experimental setup

Testbeds.

We deployed BlockRate on an indoor mesh testbed and two outdoor vehicular testbeds, referred to as MESH, VEHICULAR1, and VEHICULAR2 respectively. MESH is a wireless mesh testbed consisting of 16 nodes located in one floor of our computer science building (Figure 7). Each node is a Mac Mini computer running Linux 2.6 with a 802.11a/b/g Atheros/MadWiFi card. VEHICULAR1 is an outdoor vehicular testbed comprising of 35 public transport vehicles operating in an area of 150 square miles. Each vehicle is equipped with a Hacom OpenBrick 1GHz Intel Celeron M system running Linux 2.6 and Atheros/MadWiFi cards. The mobility of the public transport vehicles is not under our control, so in order to experiment with varying levels of mobility, we also deployed BlockRate on a private vehicular testbed, VEHICULAR2, consisting of two cars each of which is equipped with a MacBook computer and the same wireless card as the MESH nodes. All cards are configured in ad hoc mode with RTS/CTS turned off.

Sec.	Environment	Setup	Testbed
§4.2	Outdoor fast-changing	Vehicle-to-vehicle Vehicle-to-AP	VEHICULAR1 VEHICULAR2
§4.3	Indoor slow-changing	Pedestrian Static	VEHICULAR2 MESH
§4.4	Interference	Static	MESH
§4.5	Reliable transfer	Static	MESH

Table 2: A summary of the experimental setup.

Compared protocols.

We compare BlockRate against the following bitrate adaptation algorithms.

SampleRate [12], packet-based scheme that maximizes goodput by selecting the bitrate with the lowest time to (successfully) transmit a packet.

Charm [9], a packet-based scheme that leverages channel reciprocity to estimate SNR and uses SNR thresholds to select the best bitrate.

CharmBlock, a straightforward adaptation of Charm to blocks. It preserves Charm's design of using channel reciprocity to predict SNR and using SNR thresholds for selecting bitrates. However it assumes that all packets in a block have the same predicted SNR, so all packets in a block are transmitted at the same bitrate.

It is unclear how to make SampleRate work on blocks without drastically modifying its key design. Bitrate selection in SampleRate is based on per-packet transmission time including the backoff and retransmission times. In packet level protocols where a packet is repeatedly retransmitted until it is successfully received or the retransmission limit is reached, it is easy to estimate these values. However, in BlockRate where lost packets are inserted into subsequent blocks and are transmitted together with new packets, estimating the backoff time for a specific packet or the effective transmission time of an entire block is non-trivial.

Table 2 summarizes the experimental setup. For all the experiments, we use packets of size 1.5KB, and blocks of size 200 packets. All experiments measure UDP goodput except for the reliable goodput experiments in §4.5.

4.2 Outdoor vehicular mobility

We first evaluate BlockRate in fast-changing outdoor setting where communicating nodes move at vehicular speeds. We consider two typical scenarios: (1) vehicle-to-vehicle, where two vehicles communicate when they pass each other; (2) vehicle-to-AP, where a moving vehicle communicates with a static access point.

4.2.1 *Vehicle-to-vehicle* [VEHICULAR1]

We use the VEHICULAR1 testbed to evaluate the performance of the algorithms under vehicular mobility. Two vehicles communicate with each other and exchange data when they cross one another. As it is not possible to execute all the algorithms simultaneously with a single NIC, we execute each algorithm on all vehicles for one weekday and analyze the aggregate data. Although the vehicular mobility patterns vary from one day to the other, we observe from Table 3 that the number of contacts and average contact duration for all the algorithms are more or less comparable. We also note that as we are measuring the average goodput, not the volume of data transferred, the small differences in the number or average duration of contacts only introduces a weak sampling bias.

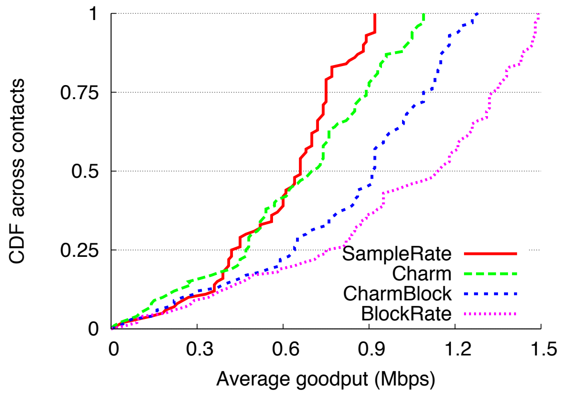


Figure 8: CDF of goodput in vehicle to vehicle experiment. BlockRate improves median goodput by 1.3× over CharmBlock and 1.6× over SampleRate and Charm.

Figure 8 shows the CDF of the goodput across all vehicle-to-vehicle contacts for each of the four algorithms. We observe that BlockRate achieves a median goodput improvement of 1.3× over CharmBlock and at least 1.6× over SampleRate and Charm. This result attests to the path loss model's ability to predict the future SNR so as to select the appropriate bitrate. In comparison, CharmBlock performs worse because it uses the same bitrate for all packets in a block, so it can not adapt to intra-block variations in channel quality. We note that CharmBlock does outperform SampleRate and Charm, demonstrating the overhead amortization benefits of blocks over packets. Taken together, these results show that the gains of amortizing overhead using blocks help all schemes (despite the reduced responsiveness), however an explicit block-based bitrate control scheme significantly outperforms packet-based bitrate control schemes used as-is with blocks.

4.2.2 *Vehicle-to-AP* [VEHICULAR2]

We use the VEHICULAR2 testbed for vehicle-to-AP experiments as the mobility (or stationarity) of the public transport vehicles in VEHICULAR1 is not under our

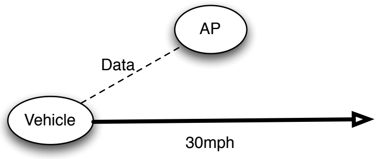


Figure 9: Vehicle-to-AP scenario.

	Number of contacts	Average contact duration
SampleRate	1719	10.2
Charm	1524	11.8
BlockCharm	1822	8.9
BlockRate	1745	9.2

Table 3: Number of contacts and average contact duration for the four algorithms in vehicle-to-vehicle experiment.

control. Furthermore, as BlockRate requires sender as well as receiver modifications, it is not feasible to experiment with open access WiFi APs on the street. So we designate one laptop in VEHICULAR2 as a stationary AP in a downtown area with dense buildings, while the other is mounted in a car moving along a straight road passing close to the AP, as shown in Figure 9. The AP continuously sends UDP packes to the vehicle as long as they are within communication range. We repeat this experiment and collect data for each of the four algorithms separately.

Figure 10(a) shows the goodput over time for the different strategies in this vehicle-to-AP setting. BlockRate improves goodput by up to 1.6× over CharmBlock and 2.8× over SampleRate and Charm. Note that BlockRate achieves the highest gains between the 5^{th} and 11^{th} seconds when it is in close proximity to the AP. This observation is consistent with the results in §3.1.2 showing that the SNR changes sharply when the vehicle is near the AP. BlockRate is able to predict this change using the path loss model and select the corresponding sequence of best bitrates across packets within a block.

Figures 10(b) and 10(c) show the bitrate selected by each scheme over time. For visual clarity, we compare BlockRate and Charm in Figure 10(b) and BlockRate and CharmBlock in Figure 10(c). SampleRate's selections are close to that of Charm and are therefore omitted. Figure 10(b) shows that BlockRate selects higher bitrates than Charm most of the time, which is consistent with the results in §2.3. Figure 10(c) shows that although CharmBlock picks similar bitrates as BlockRate during the first 5 and last 3 seconds, it is unable to predict sharp SNR changes when the vehicle is closest to the AP (between 6-11 seconds) and consistently selects lower bitrates than BlockRate.

4.3 Indoor static and pedestrian mobility

Although the case for block-based bitrate control is strongest in fast-changing conditions, for the sake of completeness, we evaluate BlockRate in a slow-changing indoor setting. We conduct experiments under (1) a walking scenario where one laptop is static and the other laptop is moved around randomly across different parts of the building, (2) a static scenario where nodes in the MESH testbed communicate with each other.

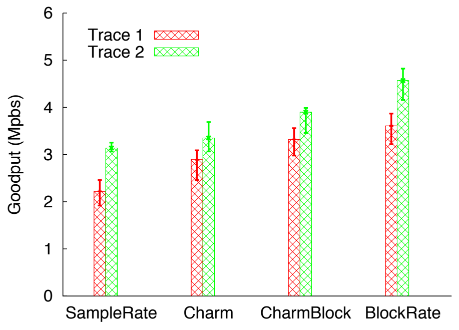


Figure 11: Goodput under pedestrian mobility. BlockRate has 1.4× better goodput than Charm.

Pedestrian mobility.

Figure 11 shows the average goodput for the walking scenario for two sets of traces. The experiments were carried out at different locations of the computer science building. The error bars show the minimum and maximum values across 5 runs of each experiment. We observe that BlockRate performs 1.1× better than CharmBlock and nearly 1.4× better than Charm and SampleRate. This experiment shows that the linear regression model works effectively under pedestrian walking speeds, even if the mobility pattern is not strictly along a straight line.

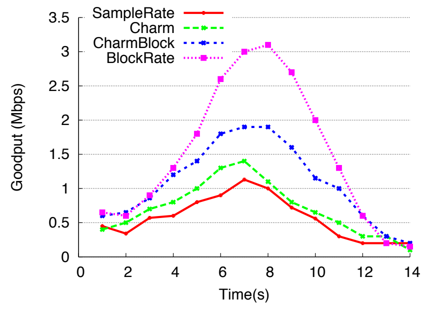
Static scenario.

We use the MESH testbed to evaluate BlockRate in a static indoor environment. We randomly select 30 links in the testbed. We choose one link at a time and continuously send UDP packets for 100 seconds and measure the goodput.

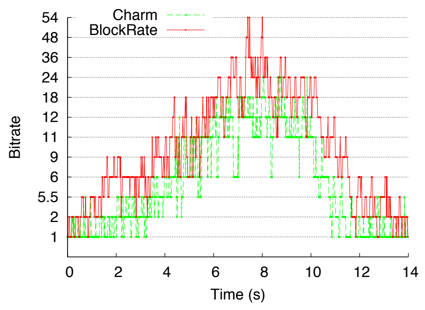
Figure 12 shows the goodput and bitrate selection results for the different algorithms. Figure 12(a) shows the CDF of goodput across 30 links. BlockRate improves median goodput by 4× over SampleRate and 2× over Charm but yields little improvement over CharmBlock because of limited variation in channel quality. Figure 12(b) shows the CDF of the bitrates selected by these algorithms across packets. BlockRate selects fairly higher bitrate than SampleRate and Charm. Comparing these results to our conclusions from §2.3 (Figure 2 and 3), we can see that BlockRate's performance is close to that of the optimal bitrate control algorithm (IdealBlock).

4.4 Impact of interference

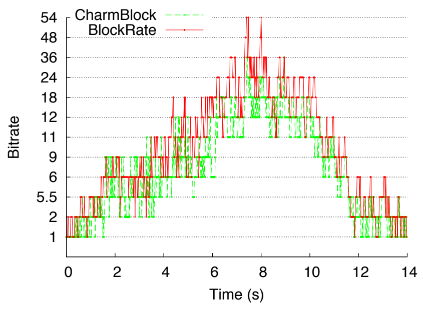
The indoor experiments described above were done in the presence of limited or no interference. In this section, we evaluate the performance of BlockRate in an



(a) Goodput

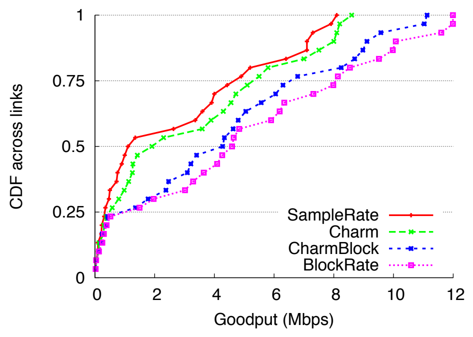


(b) Per-packet bitrate selection by BlockRate and Charm

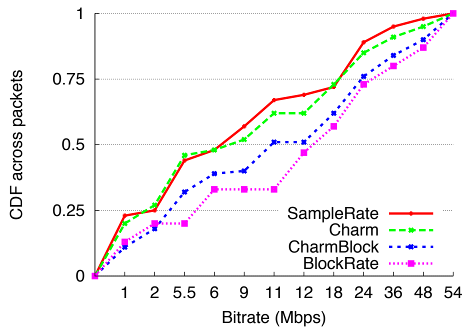


(c) Per-packet bitrate selection by BlockRate and CharmBlock

Figure 10: Goodput and bitrate selected in vehicle-to-AP experiment. BlockRate improves goodput by up to 1.6× over CharmBlock and 2.8× over SampleRate and Charm.



(a) CDF of goodput across 30 links



(b) CDF of bitrate across all packets

Figure 12: Goodput and bitrate selection results on the Mesh testbed. BlockRate improves median goodput by 2× over Charm and 4× over SampleRate.

interference-dominated environment. This experiment conducted on the MESH testbed, consists of two Access Points, each having three clients communicating with it. The nodes are selected such that they interfere with each other when transmitting data concurrently. We run each bitrate algorithm on all these nodes concurrently and measure the aggregate goodput.

Figure 13 shows the aggregate goodput achieved by the different bitrate schemes with the error bars showing the minimum and maximum values across 5 runs of each experiment. We observe that the improvement over CharmBlock is small and similar to the interference-free experiment (in Figure 11). But BlockRate has an improvement of 1.3× over Charm. As expected, we find that SampleRate performs rather poorly, primarily because SampleRate cannot distinguish between interference and poor channel quality.

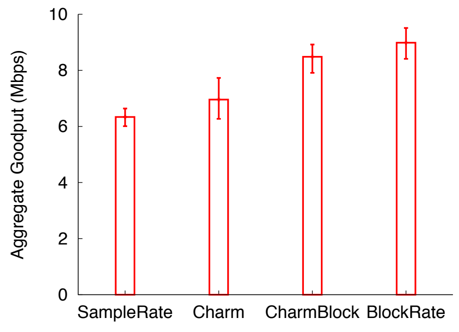


Figure 13: UDP: Goodput under interference. BlockRate has an improvement of 1.3× over Charm.

4.5 Reliable transfer performance

Although bitrate control schemes are commonly evaluated against the unreliable goodput metric (as the functionality resides strictly below the transport layer), it is critical to show that a bitrate control scheme performs well underneath a reliable transport protocol. In particular, as a reliable transport protocol imposes its own feedback-driven rate control loop atop a feedback-driven bitrate control loop at the link layer, improvements in the latter may not necessarily translate to improvements in the former.

We evaluate reliable goodput for block-based schemes under Hop, a block-based reliable transport protocol [11], and packet-based schemes under TCP. We do not present combinations of block-based bitrate control and a packet-based transport layer such as TCP since this needs to address complex interactions between the layers. It also requires a re-implementation of BlockRate within the kernel to reduce overheads when TCP's window size is smaller than a block size. These are part of our ongoing efforts. In contrast, Hop uses a hop-by-hop backpressure scheme that is well suited to large blocks. We conduct these experiments on the MESH testbed by

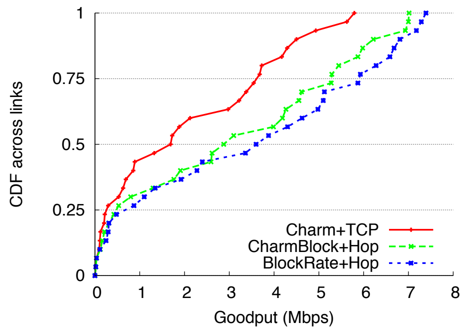


Figure 14: Goodput of reliable transfer performance in the Mesh testbed. BlockRate+Hop has an improvement of 2.1× over Charm+TCP.

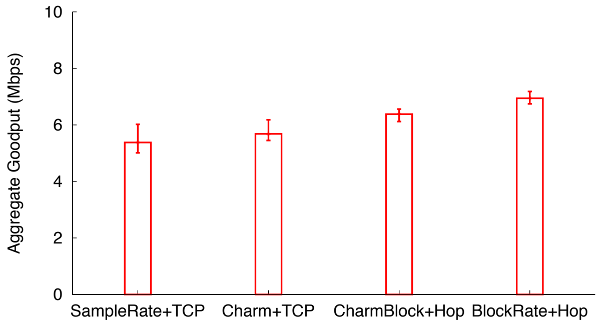


Figure 15: Reliable transfer: Goodput under interference. BlockRate+Hop outperforms Charm+TCP by 22%.

randomly selecting 30 links. Each run of the experiment lasts for 100 seconds.

Figure 14 shows the CDF of the reliable goodput across links in the testbed for each bitrate control scheme. We observe that BlockRate+Hop outperforms Charm-Block+Hop and Charm+TCP by 1.2× and 2.1× respectively with respect to the median goodput.

Figure 15 evaluates the reliable goodput of different bitrate control schemes under interference. The experimental setup is similar to §4.4. We observe that the performance of BlockRate surpasses CharmBlock by 10% and Charm by 22%.

5. RELATED WORK

BlockRate builds upon a large body of existing work in wireless bitrate control. What primarily distinguishes us is our goal, namely, to investigate the interaction of bitrate control with block transmission and to design a bitrate control algorithm optimized for blocks, which to our knowledge has not been considered before. Furthermore, we extensively evaluate BlockRate over a large-scale vehicular testbed involving naturally occurring mobility and environment patterns in addition to static or pedestrian mobility scenarios as in prior work.

To place BlockRate in relation to existing work as well as to justify the choice of protocols that we compare it against, we classify different bitrate control schemes in Table 4 along the following dimensions that we discuss in turn.

Monitored metric Existing bitrate control schemes monitor a variety of metrics to estimate channel quality including the time to (successfully) transmit a packet [12], packet loss rate [20, 13], SNR [9, 14, 8, 17, 21], and more recently PHY-layer hints, e.g., SoftRate [19] uses "SoftPHY" or confidence values conveyed by PHY to learn the bit error rate (BER), and AccuRate [18] that improves upon this scheme by monitoring symbol dispersion and using it to jump to the best bitrate in a single step. In comparison, BlockRate uses a history-trained SNR-bitrate table similar in spirit to Charm [9] but with an important difference, namely, that BlockRate is designed to use mutiple bitrates predictively before it receives any new feedback about channel quality.

Blocks vs. packets Using richer information about channel behavior such as PHY hints may further improve the performance of BlockRate, but is complementary to our primary goal of optimizing bitrate control for large blocks as opposed to small packets. PHY hints are particularly useful to react to ephemeral and fine-grained changes in channel quality, however this benefit comes at the cost of high per-packet feedback overhead. However, as we show in §2, there is much more value in amortizing this overhead even if it comes at the cost of ignoring fine-grained ephemeral changes in channel quality. In fact, as shown in Table 4, most existing bitrate control schemes are implicitly designed for per-packet feedback. One exception is MiRA [13], a scheme that is evaluated over proprietary 802.11n cards using blocks, but their focus is on optimizing bitrate control in MIMO settings that is complementary to our goals.

Mobility Although some prior works have experimented with pedestrian mobility [20, 13, 9] and others have experimented with vehicle-like mobility using simulated channel models (marked "fast-changing" in table) [19, 18, 8], we are not aware of any bitrate control scheme that has been evaluated under outdoor vehicular mobility with the sole exception of [5]. In comparison to Camp and Knightly [5] who experiment with small-scale vehicle-to-AP scenarios, our vehicular mobility experiments additionally involve thousands of vehicle-to-vehicle contacts; furthermore, these contacts are naturally occurring between public transport vehicles.

Implementation BlockRate's design as well as the choice of alternate protocols we were able to easily compare it against was dictated in part by our goal of being immediately deployable in widely used commodity wireless cards. This limited the protocols we could compare against to the first five rows in the table for which 802.11-based implementations were available. Out of these, we picked one SNR-based scheme (Charm) and one non-SNR based scheme (SampleRate). We were unable to compare agains MiRA [13] as their implementation is based on a proprietary 802.11n card, and we were unable to find an 802.11n card that exposed SNR infor-

Algorithms	**Monitored metric**	**Transport protocol**	**Mobility**	Implementation + eval.
SampleRate [12]	Txmit time	UDP	Static	802.11 NIC
RRAA [20]	Loss rate	UDP, TCP	Static, pedestrian	802.11 NIC
MiRA [13]	Loss rate	UDP, TCP, block	Static, pedestrian	802.11n NIC
BlockRate	SNR	UDP, Hop, block	Static, pedestrian, vehicular	802.11 NIC
Charm [9], SGRA [21]	SNR	UDP	Static, pedestrian (Charm)	802.11 NIC
Holland et al. [8]	SNR	UDP, TCP	Slow- and fast-changing	ns-2 simulator
CK [5]	SNR	UDP	Static, vehicular	WARP
FARA [14]	SNR	UDP	Static	WiGLAN
SoftRate [19]	SoftPHY/BER	TCP	Static, pedestrian, fast-changing	USRP traces + ns-3 simulator
AccuRate [18]	Symbol dispersion	UDP	Static, pedestrian, fast-changing	USRP + channel simulator/emulator

Table 4: Comparison of BlockRate to existing algorithms

mation and rate control hooks similar to the popular Atheros/MadWiFi combination. We recently obtained the ns-3 implementation of SotfRate from the authors (but have not yet received the channel traces used in the paper) and, for the sake of completeness, plan to evaluate in simulation the added benefit of using richer PHY information in BlockRate.

6. CONCLUSIONS

Recent trends in research as well as in practice (e.g., commodity 802.11n cards) suggest significant performance benefits of amortizing overhead across large blocks compared to small packets. However, state-of-the-art bitrate control algorithms are primarily designed to react on a per-packet basis, so they are forced to trade-off the gains of amortizing overhead using blocks against the performance loss due to the reduced responsiveness of bitrate control.

Our contribution, the design and implementation of BlockRate, is one of synthesis. We show that the benefits of blocks can be leveraged without compromising on the responsiveness of bitrate control. The key insight in BlockRate is to convert the hurdle, namely large feedback delay with blocks, to an opportunity by predicting coarse-grained channel quality trends over the course of block transmission delays. Our measurements under a variety of typically encountered vehicular and pedestrian mobility scenarios show both that it is possible to predict channel quality trends at coarse time scales and that leveraging these predictions results in significant gains in unreliable as well as reliable goodput.

7. REFERENCES

[1] Linear regression. http://en.wikipedia.org/wiki/Linear_regression.
[2] Madwifi. http://madwifi-project.org.
[3] Proceedings of the IEEE: Special Issue on Adaptive Modulation and Transmission in Wireless Systems, 2007.
[4] 802.11n: The next generation of wireless performance. Cisco White Paper, 2009.
[5] J. Camp and E. Knightly. Modulation rate adaptation in urban and vehicular environments: cross-layer implementation and experimental evaluation. MobiCom, 2008.
[6] A. Duel-Hallen. Fading channel prediction for mobile radio adaptive transmission systems. Proceedings of the IEEE, Vol. 95: pp. 2299-2313, 2007.
[7] A. Goldsmith and S. Chua. Adaptive coded modulation for fading channels. IEEE Transactions on Communication, Vol. 43: pp. 595-602, May 1998.
[8] G. Holland, N. Vaidya, and P. Bahl. A rate-adaptive mac protocol for multi-hop wireless networks. MobiCom, 2001.
[9] G. Judd, X. Wang, and P. Steenkiste. Efficient channel-aware rate adaptation in dynamic environments. MobiSys, 2008.
[10] K. LaCurts and H. Balakrishnan. Measurement and Analysis of Real-World 802.11 Mesh Networks. In *Internet Measurement Conference*, 2010.
[11] M. Li, D. Agrawal, D. Ganesan, and A. Venkataramani. Block-switched networks: a new paradigm for wireless transport. In *NSDI*, 2009.
[12] R. T. Morris, J. C. Bicket, and J. C. Bicket. Bit-rate selection in wireless networks. Technical report, Masters thesis, MIT, 2005.
[13] I. Pefkianakis, Y. Hu, S. H. Wong, H. Yang, and S. Lu. Mimo rate adaptation in 802.11n wireless networks. MobiCom, 2010.
[14] H. Rahul, F. Edalat, D. Katabi, and C. Sodini. Frequency-Aware Rate Adaptation and MAC Protocols. In *MOBICOM*, 2009.
[15] T. Rappaport. *Wireless Communications: Principles and Practice*. 2nd edition, 2001.
[16] L. Ravindranath, C. Newport, H. Balakrishnan, and S. Madden. Improving wireless network performance using sensor hints. In *NSDI*, 2011.
[17] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knightly. Opportunistic media access for multirate ad hoc networks. MobiCom, 2002.
[18] S. Sen, N. Santhapuri, R. R. Choudhury, and S. Nelakuditi. Accurate: constellation based rate estimation in wireless networks. In *NSDI*, 2010.
[19] M. Vutukuru, H. Balakrishnan, and K. Jamieson. Cross-Layer Wireless Bit Rate Adaptation. In *ACM SIGCOMM*, 2009.
[20] S. H. Y. Wong, H. Yang, S. Lu, and V. Bharghavan. Robust rate adaptation for 802.11 wireless networks. In *Mobicom*, 2006.
[21] J. Zhang, K. Tan, J. Zhao, H. Wu, and Y. Zhang. A practical snr-guided rate adaptation. Infocom, 2008.