Rethinking Congestion Control for Cellular Networks

Prateesh Goyal, Mohammad Alizadeh, Hari Balakrishnan
MIT Computer Science and Artificial Intelligence Laboratory
{prateesh,alizadeh,hari}@csail.mit.edu

ABSTRACT

We propose Accel-Brake Control (ABC), a protocol that integrates a simple and deployable signaling scheme at cellular base stations with an endpoint mechanism to respond to these signals. The key idea is for the base station to enable each sender to achieve a computed target rate by marking each packet with an "accelerate" or "brake" notification, which causes the sender to either slightly increase or slightly reduce its congestion window. ABC is designed to rapidly acquire any capacity that opens up, a common occurrence in cellular networks, while responding promptly to congestion. It is also incrementally deployable using existing ECN infrastructure and can co-exist with legacy ECN routers. Preliminary results obtained over cellular network traces show that ABC outperforms prior approaches significantly.

1 INTRODUCTION

This paper introduces a new router-signaling and endpoint protocol to determine transmission rates in cellular networks. These networks exhibit significant variations in bottleneck link rates even over short time durations [30]. One would therefore expect that an active queue management (AQM) scheme running at the cellular bottleneck router,[1] sending explicit congestion notification (ECN) [7] singals to senders using *direct knowledge of the time-varying link capacity*, would achieve higher throughput and lower delays than end-to-end approaches. Yet, recent work has shown that schemes like Cubic-over-sfqCoDel [21] do not significantly outperform end-to-end methods like Sprout [30] and Verus [33].

We hypothesize, and substantiate in this paper, that a well-designed signaling scheme running at the cellular bottleneck link, working in concert with a suitable endpoint algorithm, can out-perform prior end-to-end and network-assisted (AQM) methods. Our hypothesis is based on two key insights. First, unlike end-to-end approaches, a router-based scheme can directly estimate link capacity, without relying on end-to-end bandwidth measurements which fundamentally require some queue buildup; on cellular networks, this queue buildup can add significant delays when link capacity decreases. Second, existing ECN-based AQM schemes send indications only when the sender should reduce its rate, but not when an increase is possible. The problem in cellular networks is that the rate can increase quickly, faster than endpoints (particularly with linear or even Cubic-like increase) can discover solely via end-to-end methods.

Most known mechanisms to signal the ability to increase a rate, such as explicit flow control in ATM networks [15], XCP [16], and RCP [27], are verbose; i.e., they require multi-bit per-packet feedback signals from the routers. Such schemes have proved hard to deploy because they are invasive, requiring changes to a component of the Internet architecture most difficult to change: the IP layer and the IP header fields. (An exception to verbosity is VCP [31], which we describe and compare against later.)

Is it possible to use just *one bit* of feedback per ACK to achieve performance comparable to verbose feedback that tells the senders transmission rates (or windows) explicitly in each ACK? We show that the answer is "yes" by developing a simple, one-bit signaling strategy that combines the richness of a fully explicit protocol with the simplicity and deployability of the ECN signaling mechanism.

Our proposal has three key ideas, two conceptual and one concerning deployment. First, the base station marks each packet with one bit of feedback corresponding to either *accelerate* or *brake* using a measured estimate of the current rate for the user and a computed estimate of a desired target rate.[2] Second, upon receiving this feedback via an ACK from the receiver (using the same concept as in ECN), the sender either accelerates its transmission by sending *two packets* (on accelerate), or decelerates by not sending any packet (on brake). This simple mechanism, which we call *Accel-Brake Control (ABC)* achieves a large dynamic range of rates; it allows the transmission rate to vary from twice the current value down to 0 within one RTT.

An important goal of our proposal is *deployability*. We show how to reuse the existing ECN infrastructure to implement ABC, presenting two deployment options for networks without and with legacy ECN routers. The first case applies

[1] We use the terms "base station" and "bottleneck router" interchangeably. We expect the signaling method to be implemented in any network element within the cellular network infrastructure that is a potential bottleneck. For transmissions *from* mobile devices, the bottleneck router is the device's network layer.

HotNets-XVI, November 30-December 1, 2017, Palo Alto, CA, USA

ACM ISBN 978-1-4503-5569-8/17/11…$15.00
https://doi.org/10.1145/3152434.3152437

[2] Maintaining the current rate and target rate requires per-user state, but cellular networks already have per-user state as base stations isolate traffic for different users in separate queues [30].

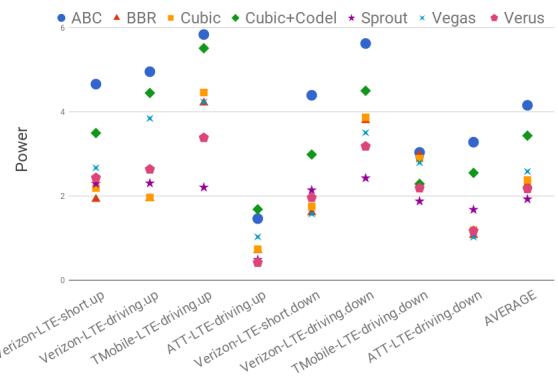


Figure 1: Power, defined as the ratio of the utilization (i.e., measured throughput divided by maximum throughput) to the 95^{th} percentile per-packet delay, of various schemes on traces from eight different cellular networks. Averaging the normalized power across the networks, ABC outperforms the second-best scheme (Cubic + Codel) by 21%.

to many cellular networks that split TCP connections at the network boundary [24, 29]; here, we show how to deploy ABC without any changes to TCP receivers (e.g., running on mobile phones). In the second case, we show how ABC can co-exist with legacy ECN with simple changes to the receiver.

Figure 1 shows preliminary performance results. ABC outperforms prior approaches in terms of throughput/delay ("power") across a variety of cellular networks. These results substantiate our position that, to obtain high throughput and low packet delays in cellular access networks, the base station should send direct rate increase/decrease feedback based on its computed estimate of time-varying capacity, and that such feedback is readily deployable using existing ECN signaling mechanisms. We hope that our results will be a call to arms to cellular network operators to adopt these simple ideas to improve user experience.

2 RELATED WORK

Traditional end-to-end protocols like TCP Cubic [9] and NewReno [10, 13] are unable to track time-varying wireless link capacities well. To achieve high throughput for these protocols, cellular networks often use deep buffers, which leads to significant bufferbloat.

Bufferbloat can be reduced using AQM schemes like CoDel [21] or PIE [22]. Although these schemes are able to achieve low packet delays, they suffer from under-utilization because of the conservative increase rules in Cubic and NewReno—when link capacity increases, the sender is unable to increase its rates quickly enough. Perhaps for this reason, we see little evidence of cellular networks using AQM schemes.

Without AQM, Cubic and NewReno rely only on packet drops to infer congestion, but (especially with deep packet buffers) this signal is too infrequent for efficient adaptation to changing link conditions. Recent end-to-end proposals like Sprout [30] and Verus [33] overcome the spareness of packet drops by using information about the RTT and the send/receive rate, and combining this information with prediction strategies to deduce available capacity.

These schemes face a trade-off between utilization (throughput) and packet delay because estimating the available capacity by end-to-end means requires some queue build up. These schemes have no information about available capacity when the queue is empty; during such periods, the best they can do is to increase the rate in "blind" fashion. This approach is problematic in networks with a large dynamic range of rates: if the increase is slow, throughput is low, but making the increase too fast inevitably causes overshoots and large queuing delays. Thus, we conclude that, end-to-end schemes will find it difficult to track the available capacity accurately. Router assistance using methods deployed at the base station (or the mobile device's network layer) are therefore important to consider.

Perhaps closest in spirit to ABC is VCP [31], which also relies on succinct feedback. In VCP, the router tells the sender what *kind* of window update to use depending on load: multiplicative-decrease, multiplicative-increase, or additive-increase. This coarse-grained feedback strategy limits VCP's effectiveness in cellular networks with a large dynamic range of rates. For example, the VCP paper uses a multiplicative increase factor of $\xi = 0.0625$, requiring 12 RTTs to double the rate. By contrast, ABC can adjust the rate between $2\times$ the current value and 0 within 1 RTT, because it distributes the feedback over a *stream* of ACKs, instead of using ACKs in isolation. Moreover, VCP is incompatible with legacy ECN, making it hard to deploy.

Recent proposals [14, 19] have proposed to use the LTE infrastructure for inferring the underlying link capacity. CQIC [19] proposes using the physical layer information at the receiver (e.g., physical resource blocks allocated) for estimating the link capacity. Unlike ABC, this approach requires receiver modification. MTG [14] proposes modifying the base station to explicitly communicate the rate, but unlike ABC, it relies on sending this rate feedback via a new TCP option [23]. However, using a TCP option in this way runs the risk of packets being dropped silently by middleboxes [12], and cannot work when IPSec encryption is used [25]. This proposal also lacks a mechanism for apportioning the bandwidth among multiple flows of a user. In §5 we show that ABC's single-bit feedback is sufficient for communicating the target rate and its performance is comparable to a full-rate-feedback variant.

3 DESIGN

In ABC, the bottleneck router provides one bit of feedback per packet. Each bit is an indication to either slightly increase (accelerate) or slightly decrease (brake) the sender's congestion window. The router determines the feedback for each packet using its estimates of a desired target rate, the current dequeue rate for the sender's packets, and the packet delays experienced by the sender's packets in the bottleneck queue.

3.1 ABC Sender

Sender's window update rule: On receiving an "accelerate" bit in an ACK, the sender sends two packets; one replaces the acknowledged packet, and the other because the congestion window has "increased by one". On receiving a "brake" feedback, the sender sends nothing, not even to replace the

packet that has been received; i.e., the congestion window reduces by one packet. The intuition for ABC can be drawn from driving a car on a busy street, where the driver's actions are to either slightly accelerate or to tap on the brake. Using a sequence of these two actions in quick succession, the driver is able to maintain a suitable car speed.

A single bit of feedback per packet is quite expressive when accumulated over several packets, such as an RTT's worth. If the sender's window is w and a fraction f of the packets are marked with "accelerate," then in the next RTT the sender will receive w ACKs,[3] of which wf will be accelerates and $w-wf$ will be brakes. As a result, the congestion window after 1 RTT will be $w+wf-(w-wf)=2wf$.

The router selects f; $f=0$ throttles the transmission entirely, $f=1/2$ retains the current window, $f=1$ doubles it, and so on. The set of achievable window changes for the next RTT depends on the number of packets in the current window; the larger this number, the finer the granularity.

Initial conditions: Each packet transmitted by the sender has the "accelerate" bit on by default. A router may change an "accelerate" to "brake" but not vice versa. Thus, by default, the behavior corresponds to $f=1$, but that situation may not last for long if the current dequeue rate approaches or exceeds the target rate, or if queues start to build at the router. Routers are not allowed to change a packet with "brake" set on it to "accelerate"; the reason is that an ABC router can unilaterally decide to slow down the rate (brake), but acceleration requires consensus along the path. We discuss the operation on paths with multiple ABC routers below.

3.2 ABC Router

The target rate: The ABC router determines the *target rate*, $tr(t)$, for each user using the current per-user link capacity, $\mu(t)$, and queueing delay, $x(t)$, using this rule:

$$tr(t)=\eta\mu(t)-\beta\frac{\mu(t)}{\delta}(x(t)-d_t)^+. \quad (1)$$

This control rule is inspired by ideas in prior work. First, as in TUB from ATM networks [15], AVQ [18], and HULL [1], ABC sets $tr(t)$ to a value $\eta\mu(t)$, where $\eta<1$ represents the target utilization. This term allows the sender to send traffic just under bottleneck capacity, trading off a little throughput for more significant delay reductions. This approach works in ABC because it doesn't need to fill up queues to accurately estimate the available capacity for each sender.

The above target rate by itself is insufficient because there is a delay of one RTT in feedback to the sender. This delayed feedback combined with inaccuracies in measurements can lead to transient queues. We introduce a second term that uses the current queuing delay, $x(t)$, to account for stagnant queue build-up. The router reduces the target rate if $x(t)$ exceeds a delay threshold, d_t (the notation y^+ is defined as $\max(y,0)$). The reason for a non-zero threshold delay is that wireless networks exhibit non-deterministic packet service times because their link rates vary, so any packet arrival process will experience *some* queue build-up. A non- zero delay threshold d_t ensures that the target rate does not react to minor delay fluctuations. The value of d_t could, in principle, change with time, but ABC currently uses a static value for simplicity.

The target rate calculation in Eq. (1) requires information about the per-user link capacity, $\mu(t)$. In cellular networks, because the scheduler manages all the traffic, it knows exactly what resources have been allocated to each user and can thus calculate $\mu(t)$. The scheduler also guarantees fairness between users while allocating resources; ABC does not affect inter-user fairness. We discuss how ABC can achieve fairness between flows of a given user in §6; this requires a small modification to the ABC sender.

Equation (1) has three constants, η,β, and δ, which govern the stability of the scheme. δ should be on the order of one RTT; since the router does not know the RTT precisely, we use 100 ms in our tests. β governs how quickly queues drain. We do not explicitly establish the stability conditions of Eq. (1) here, but prior methods [3, 16] apply.

Packet marking: To achieve the target rate, $tr(t)$, the router measures the sender's current dequeue rate, $cr(t)$, and calculates the fraction of packets, $f(t)$, that remain marked as "accelerate" using the following rule:

$$f(t)=\min\Big\{\frac{1}{2}\cdot\frac{tr(t)}{cr(t)},1\Big\}. \quad (2)$$

The ABC router ensures that no more than a fraction $f(t)$ of packets have the "accelerate" indication set. The pseudocode for the packet marking algorithm is shown below.

```
On each idle period with no packets:
spare_accel = 0;
In steady state:
for each packet do
    calculate f(t) using Eq. (2);
    spare_accel = spare_accel + f(t);
    if packet marked with accelerate then
        if spare_accel > 1 then
            spare_accel = spare_accel − 1;
        else
            mark brake;
        end
    end
    spare_accel = min(spare_accel, MAXSPARE);
    // MAXSPARE is a constant, like 5
end
```

Algorithm 1: Packet marking at an ABC router. The MAXSPARE parameter (set to 5 in our implementation) limits the burstiness of "accelerate" signals.

In this algorithm, packet marking is deterministic; it is also possible to mark probabilistically, but the results will be burstier.

Handling multiple access links: Thus far we have described our solution for a single access link setting, but there can be multiple access links on a path; for example, a cellular uplink

[3] Assuming no delayed ACKs; with delayed ACKs and appropriate byte counting (RFC 3465), a slightly different calculation will yield the same result.

and a cellular downlink in a video conference between two smartphone users.

In such settings, the sender should send at the smaller of the two target rates, which implies that the final "accelerate" fraction received by the sender should be the minimum over the multiple ABC links. The packet marking mechanism described above automatically achieves this behavior because access routers can only convert an accelerate to a brake.

Rate measurement timescales: We use a moving window of time to calculate the target rate and current dequeue rate. When the access link's rate varies quickly, using a shorter time window gives better performance, but if the time window is too short, then the rate estimates might be incorrect. For cellular networks, the time window used for measurements should be at least the average inter-schedule time of a user at the router. Thus, at a base station, the time window should be greater than $\frac{N \cdot S}{M}$, where N is the number of active users, S is the sub-frame size ($\approx$ 1 ms for LTE), and M is the number of "resource blocks" per sub-frame. We use a window size of 20 ms in our experiments over traces.

4 DEPLOYMENT WITH ECN

We now describe two deployment options for ABC. We begin with a brief review of standard ECN [8], which our proposal builds on. Then we discuss how to deploy ABC in networks without legacy ECN in the routers. In this case, we can deploy ABC without any modifications to TCP receivers, i.e., mobile phones can take advantage of ABC for download traffic without any software changes.[4] Finally, we present a way to deploy ABC that is compatible with legacy ECN but requires a simple modification to TCP receivers.

Standard ECN: IP packets have two ECN-related bits: ECT and CE. These two bits are traditionally interpreted as follows

ECT	CE	Interpretation
0	0	Non ECN-Capable Transport
0	1	ECN-Capable Transport ECT(1)
1	0	ECN-Capable Transport ECT(0)
1	1	ECN set

Routers interpret both (01) and (10) for these bits to indicate that the connection is ECN capable. To mark a packet with ECN, a router changes the bits to (11). Upon receiving an ECN mark (11), the receiver sets the *ECN Echo (ECE)* flag in *all* subsequent ACKs until it receives a *Congestion Window Reduced (CWR)* notification from the sender, i.e., a packet with the CWR flag in the TCP header set to 1. Thus receivers echo ECN marks for a full round-trip worth of ACKs.

No legacy ECN: Many cellular networks use *TCP proxies* to split connections at the network boundary [24, 29]. The operators of such networks can easily disable ECN on the routers inside the cellular network. This allows a straight forward deployment of ABC without receiver modification. ABC senders simply use either (01) or (10) for *accelerate* and routers use (11) to indicate *brake*. To ensure that the receiver conveys accelerate/brake signals accurately, the sender sets the CWR flag on *every* packet. This ensures that the receiver echoes a brake exactly once (instead of for a full RTT), assuming all packets are acknowledged and ACKs are not lost. More accurate ECN feedback mechanisms in the presence of delayed/lost ACKs have been proposed [2], which require receiver modifications. ABC can also leverage these techniques.

With legacy ECN: ABC can coexist with legacy ECN by reinterpreting the ECT and CE bits as follows

ECT	CE	Interpretation
0	0	Non ECN-Capable Transport
0	1	*accelerate* ECT(1)
1	0	*brake* ECT(0)
1	1	ECN set

The ABC sender begins with (01) on every packet (*accelerate*). An ABC router can signal *brake* by flipping the bits to (10). For legacy routers, both (01) and (10) are equivalent and indicate ECN capable transport. Thus a legacy routers can simply use (11) to signal congestion, exactly as in standard ECN. Finally, the receiver can distinguish between accelerate (01), brake (10) and ECN set (11).

The receiver must convey both standard ECN and accelerate/brake for ABC. We can use the ECE flag for ECN feedback. For ABC feedback, we can reuse the NS (nonce sum). The NS bit was originally proposed to ensure integrity of ECN feedback [6], but this feature has since been reclassified as historic [2] since it was never deployed. Therefore, we can repurpose the NS bit for ABC's feedback. Using this scheme, the sender can react to drops/ECN marks same as standard TCP.

ABC routers do not interfere with the proper operation of ECN for standard TCP flows. All TCP implementations transmit packets with ECT/CE set to (10) to indicate ECN-capable transport. Since ABC only converts "accelerate" (01) to "brake" (10), it will ignore any packet marked with (10). As a result, there is no need for senders to convey ABC capability in packet headers.

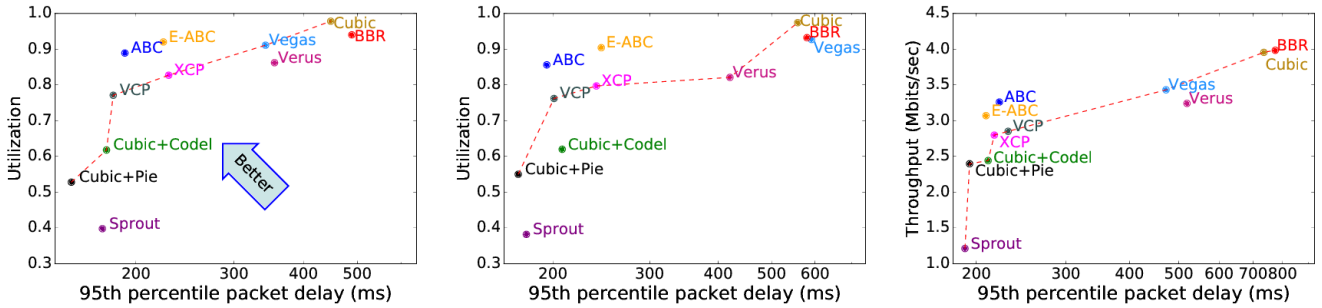
Note that any internal encryption used by the cellular network provider does not hinder the ABC router's ability to modify ECN bits. The ABC router (base station or mobile phone) is typically the location where any form of in-network encryption is removed, which means that the ABC router will have access to the unencrypted packets and can modify the ECN bits.

A potential vulnerability of the above scheme is that a malicious router on the path can convert a brake to an accelerate. However, we can easily detect this at the senders by sending packets marked with brake once in a while; if any of these packets return with accelerate, then there are malicious entities on the path.

5 PRELIMINARY EXPERIMENTS

We evaluate ABC by running real cellular link traces using Mahimahi, a network emulator tool [20]. We compare ABC to two state-of-the-art end-to-end protocols for cellular networks, Sprout [30] and Verus [33]; two AQM schemes, Cubic+Codel [21] and Cubic+PIE [22]; two explicit congestion control protocols, XCP [16] and VCP [31]; and three end-to-end protocols, Cubic [9], Vegas [4], and BBR [5]. In each

[4]The transport stack on mobile phones of course needs to change to use ABC for upload traffic.



(a) Uplink **(b) Downlink** **(c) Uplink + Downlink**

Figure 2: ABC vs. previous schemes on three Verizon cellular network traces. In each case, ABC outperforms all other schemes and sits well outside the Pareto frontier of previous schemes.

experiment, we start one flow and measure utilization/throughout, and the mean, median, and 95^{th} percentile (p95) per-packet delay statistics. We use $\eta = 0.98, \beta = 0.75, \delta = 100$ ms and $d_t = 50$ ms for ABC parameters in all experiments.

Figure 1 previously showed the ratio of throughput to 95^{th} percentile packet delay ("power") across eight cellular traces. ABC outperforms previous schemes, with improvements in power of up to 47%. Figure 2 shows the throughput and 95^{th} percentile packet delay separately for three network traces, representing uplink, downlink, and combined uplink/downlink scenarios. In the combined case, we ran traffic with the uplink and downlink connected in series.

We make five observations from these experiments. First, ABC exhibits a substantially better throughput/delay tradeoff than previous schemes. The dashed lines in each figure show the "Pareto frontier" for previous approaches; each scheme on this frontier achieves higher throughput or lower delay than the others. ABC sits well outside the best previous Pareto frontier.

Second, ABC outperforms even previous router-assisted schemes like XCP that include verbose feedback. The reason is that XCP's control law is not suitable for cellular networks. It uses the "persistent queue size," measured as the *minimum* queue size in the last control interval, to calculate the feedback. However, the minimum queue size is an unreliable signal in cellular networks, because of the large variations in link capacity (and queue size).

Third, ABC also outperforms VCP [31], a previous scheme with succinct feedback. As discussed in §2, VCP is restricted to select among a few hard-coded update rules, such as a fixed multiplicative or additive increase. By not hard-coding a fixed increase rule, ABC performs better: across the eight cellular network traces, ABC achieved a 15% higher utilization (0.82 v. 0.72), incurring only a 6% higher mean packet delay (145 ms v. 135 ms).

Fourth, ABC's performance is similar to an explicit variant of ABC (E-ABC) that uses multiple bits to simply specify the target rate on every packet. In contrast to ABC, because this version of E-ABC is rate-based, it is inherently slower in reacting to congestion. We find that E-ABC achieves slightly higher throughput and slightly higher latency compared to ABC. The important conclusion is that ABC's single-bit feedback, with its deployment advantages, does not leave performance "on the table".

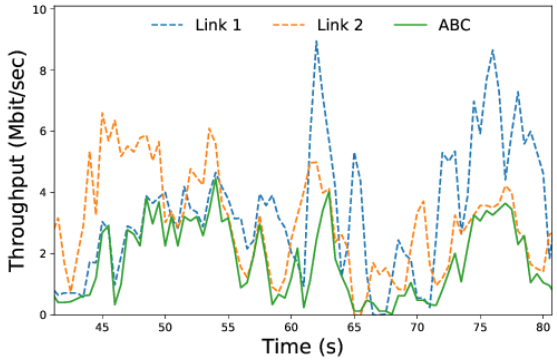


Figure 3: ABC on a path with two cellular links. ABC's rate tracks the bottleneck closely.

Finally, as Figure 2c shows, ABC continues to work well in the presence of multiple bottlenecks. Figure 3 shows an example of ABC on a path with two cellular links. At any given time, one of the two links is the bottleneck, and ABC's rate tracks the bottleneck closely.

6 FUTURE WORK

Estimating the link capacity at base stations: Our proposal relies on the ability of the base station to estimate, $\mu(t)$, the available per-user link capacity (see Eq. (1)). In our experiments, we obtained this information by measuring the rate of packet transmissions in our cellular traces. However, these traces were obtained using a traffic generator which saturated the cellular link [30]. In practice, during periods where the queue for a user is empty or the occupancy is very low, scheduling algorithms on the base station (e.g., the proportionally fair scheduler [26, 28]) may allocate less channel time to the user than it *could* have achieved had it sent more traffic [17]. Therefore, naïvely using the scheduler's allocation as $\mu(t)$ may cause some underutilization, particularly since ABC tries to maintain small queues by design.

Fortunately, the base station has low-level visibility into physical layer resources (e.g., physical resource blocks and the modulations for each user), from which it can estimate the user's *potential* rate if it sends enough traffic to be bottlenecked at the base station. For example, a simple approach could be to simulate the proportional fair scheduler with the current conditions but assuming the user has a large backlog. We therefore

believe it should be possible to obtain a good estimate of $\mu(t)$, even during periods of low queue occupancy. We plan to investigate the details of a method for this purpose in future work.

Co-design of base station schedulers and ABC signaling: As mentioned above, existing scheduling algorithms for base stations use the queue backlog as a proxy for the user's demand [17]. ABC signaling mechanisms may provide a better way to estimate user demands at the base station to improve scheduling decisions. In particular, ABC senders can be modified to express their demand using "accelerate" marks: rather than marking all packets with "accelerate," a sender can indicate its desired rate increase by marking an appropriate fraction of packets "accelerate".[5] The base station can use this information to both make scheduling decisions and to calculate the target rate for ABC.

Fairness among ABC flows: Cellular networks ensure user-level fairness by scheduling users out of separate queues, but achieving flow-level fairness for a given user requires changes to ABC's design. ABC's window update rule is effectively a multiplicative-increase/multiplicative-decrease (MIMD) strategy (§3), which does not provide fairness among flows contending on a link (see Figure 4a for an illustration).

A simple modification to the ABC sender promises to be very effective. We can simply add an *additive-increase* (AI) component to the window update rule; e.g., the sender can increase the congestion window by 1 every T ms, in addition to reacting to accelerate/brake feedback. This idea, inspired by traditional additive-increase/multiplicative-decrease (AIMD) congestion control protocols, works because ABC's brakes reduce the window via multiplicative-decrease during congestion. Adding an additive component thus creates an AIMD-like behavior. Figure 4b shows how with an AI component, two ABC flows achieve the same throughput.

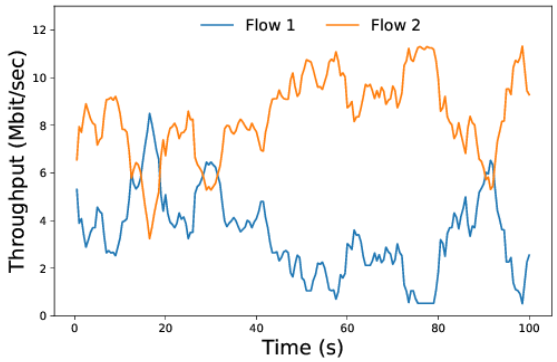
A potential drawback of this approach is that the ABC router will receive more traffic than it anticipates. This can cause persistent queues and delay, particularly, if a user has a large number of flows. We plan to investigate ideas like adding a penalty based on the integral of the queuing delay to the target rate calculation (similar to the PI [11] AQM) to solve this problem.

Using ABC in other access networks and applications: ABC is likely to be beneficial in other access networks besides cellular networks (e.g., home broadband, DSL, etc.). Also, ABC's ability to accurately estimate the network's underlying bandwidth without filling up queues can be very useful for applications like live video streaming, where we want to adapt the video resolution to network conditions while maintaining low latency. Previous work [32] has shown that estimating the underlying bandwidth accurately can increase user perceived quality of experience on LTE networks. We plan to explore such use cases in future work.

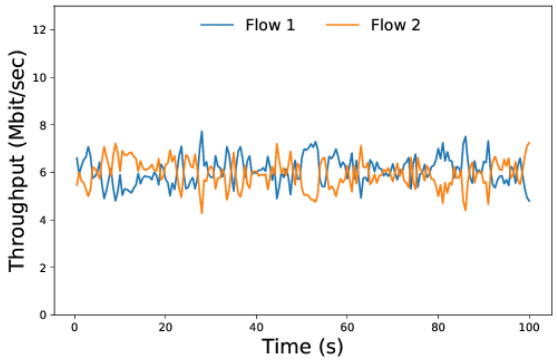
7 CONCLUSION

We presented ABC, a new approach to congestion control in cellular networks in which bottleneck routers signal rate increases and decreases to senders based on a measured estimate of the current rate and an explicitly computed target rate. ABC quickly and accurately adapts to time-varying cellular link conditions, despite using a very simple signaling scheme that requires only 1 bit of feedback per acknowledgment. Our preliminary results showed that ABC outperforms prior approaches, including state-of-the-art cellular congestion protocols, AQM schemes, explicit rate control protocols, and more. Across eight real cellular network traces, ABC improves "power" (ratio of throughput to 95^{th} percentile packet delay) by 21% on average. ABC is also deployable immediately using existing ECN infrastructure and is backwards compatible with legacy ECN routers and traffic.

Our work shows that significant improvements to congestion control in cellular networks are both possible and practical; we hope that our results will encourage cellular network operators to adopt congestion signaling strategies like ABC to improve user experience.



(a) ABC (design in §3)



(b) ABC + AI

Figure 4: Fairness between two ABC flows competing on a single link with a fixed capacity of 12 Mbit/sec. The additive-increase (AI) component leads to fairness among flows.

ACKNOWLEDGMENTS

This work was funded in part by NSF award 1407470 and DARPA award HR0011-15-2-0047. We thank Anirudh Sivaraman, Peter Iannucci, Ravi Netravali, Srinivas Narayana, and the HotNets reviewers for their useful comments and feedback.

[5]The sender can express a demand of d packets/sec by setting "accelerate" on d/2 packets every second.

REFERENCES

[1] M. Alizadeh, A. Kabbani, T. Edsall, B. Prabhakar, A. Vahdat, and M. Yasuda. Less is more: trading a little bandwidth for ultra-low latency in the data center. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*, pages 19–19. USENIX Association, 2012.

[2] B. B., K. M., and S. R. More Accurate ECN Feedback in TCP. https://tools.ietf.org/html/draft-ietf-tcpm-accurate-ecn-03, 2017.

[3] H. Balakrishnan, N. Dukkipati, N. McKeown, and C. J. Tomlin. Stability analysis of explicit congestion control protocols. *IEEE Communications Letters*, 11(10), 2007.

[4] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson. TCP Vegas: New Techniques for Congestion Detection and Avoidance. In *SIGCOMM*, 1994.

[5] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson. Bbr: Congestion-based congestion control. *Queue*, 14(5):50:20–50:53, Oct. 2016.

[6] D. Ely, N. Spring, D. Wetherall, S. Savage, and T. Anderson. Robust congestion signaling. In *Network Protocols, 2001. Ninth International Conference on*, pages 332–341. IEEE, 2001.

[7] S. Floyd. TCP and Explicit Congestion Notification. *CCR*, 24(5), Oct. 1994.

[8] S. Floyd, K. Ramakrishnan, and D. L. Black. RFC 3168: The Addition of Explicit Congestion Notification (ECN) to IP. https://tools.ietf.org/html/rfc3168, 2001.

[9] S. Ha, I. Rhee, and L. Xu. CUBIC: A New TCP-Friendly High-Speed TCP Variant. *ACM SIGOPS Operating System Review*, 42(5):64–74, July 2008.

[10] J. C. Hoe. Improving the Start-up Behavior of a Congestion Control Scheme for TCP. In *SIGCOMM*, 1996.

[11] C. V. Hollot, V. Misra, D. Towsley, and W.-B. Gong. On designing improved controllers for aqm routers supporting tcp flows. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1726–1734. IEEE, 2001.

[12] M. Honda, Y. Nishida, C. Raiciu, A. Greenhalgh, M. Handley, and H. Tokuda. Is it still possible to extend tcp? In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 181–194. ACM, 2011.

[13] V. Jacobson. Congestion Avoidance and Control. In *SIGCOMM*, 1988.

[14] A. Jain, A. Terzis, H. Flinck, N. Sprecher, S. Arunachalam, and K. Smith. Mobile throughput guidance inband signaling protocol. *IETF, work in progress*, 2015.

[15] R. Jain. Congestion control and traffic management in atm networks: Recent advances and a survey. *Computer Networks and ISDN systems*, 28(13):1723–1738, 1996.

[16] D. Katabi, M. Handley, and C. Rohrs. Congestion Control for High Bandwidth-Delay Product Networks. In *SIGCOMM*, 2002.

[17] T. E. Kolding. Link and system performance aspects of proportional fair scheduling in wcdma/hsdpa. In *Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th*, volume 3, pages 1717–1722. IEEE, 2003.

[18] S. Kunniyur and R. Srikant. Analysis and Design of an Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management. In *SIGCOMM*, 2001.

[19] F. Lu, H. Du, A. Jain, G. M. Voelker, A. C. Snoeren, and A. Terzis. Cqic: Revisiting cross-layer congestion control for cellular networks. In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, pages 45–50. ACM, 2015.

[20] R. Netravali, A. Sivaraman, S. Das, A. Goyal, K. Winstein, J. Mickens, and H. Balakrishnan. Mahimahi: Accurate Record-and-Replay for HTTP. In *USENIX Annual Technical Conference*, pages 417–429, 2015.

[21] K. Nichols and V. Jacobson. Controlling Queue Delay. *ACM Queue*, 10(5), May 2012.

[22] R. Pan, P. Natarajan, C. Piglione, M. Prabhu, V. Subramanian, F. Baker, and B. VerSteeg. Pie: A lightweight control scheme to address the bufferbloat problem. In *14th International Conference on High Performance Switching and Routing (HPSR)*, 2013.

[23] J. Postel. Transmission control protocol. 1981.

[24] L. Ravindranath, J. Padhye, R. Mahajan, and H. Balakrishnan. Timecard: Controlling user-perceived delays in server-based mobile applications. In *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*, pages 85–100. ACM, 2013.

[25] K. Seo and S. Kent. Security architecture for the internet protocol. 2005.

[26] S. Sesia, M. Baker, and I. Toufik. *LTE-the UMTS long term evolution: from theory to practice*. John Wiley & Sons, 2011.

[27] C. Tai, J. Zhu, and N. Dukkipati. Making Large Scale Deployment of RCP Practical for Real Networks. In *INFOCOM*, 2008.

[28] D. Tse and P. Viswanath. *Fundamentals of wireless communication*. Cambridge university press, 2005.

[29] Z. Wang, Z. Qian, Q. Xu, Z. Mao, and M. Zhang. An untold story of middleboxes in cellular networks. In *ACM SIGCOMM Computer Communication Review*, volume 41, pages 374–385. ACM, 2011.

[30] K. Winstein, A. Sivaraman, and H. Balakrishnan. Stochastic Forecasts Achieve High Throughput and Low Delay over Cellular Networks. In *NSDI*, 2013.

[31] Y. Xia, L. Subramanian, I. Stoica, and S. Kalyanaraman. One More Bit is Enough. *IEEE/ACM Trans. on Networking*, 16(6):1281–1294, 2008.

[32] X. Xie, X. Zhang, S. Kumar, and L. E. Li. pistream: Physical layer informed adaptive video streaming over lte. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, pages 413–425. ACM, 2015.

[33] Y. Zaki, T. Pötsch, J. Chen, L. Subramanian, and C. Görg. Adaptive congestion control for unpredictable cellular networks. *SIGCOMM Computer Communication Review*, 45(4):509–522, 2015.