# Green With Envy: Unfair Congestion Control Algorithms Can Be More Energy Efficient

Serhat Arslan
Stanford University
sarslan@stanford.edu

Sundararajan Renganathan
Stanford University
rsundar@stanford.edu

Bruce Spang
Stanford University
bspang@stanford.edu

## ABSTRACT

Despite 40 years of active research on congestion control, there has been little or no consideration of how it impacts the energy usage of end-hosts or networking equipment. Particularly with the burgeoning energy consumption of data centers and wide-area networks, we argue that the time is ripe for the networking community to start thinking along these lines. To pave the way, we conduct lab experiments to measure the energy used by popular congestion control algorithms. We consider various aspects of congestion control and the rich research challenges that arise when we consider energy efficiency. Specifically, we find that fairness for the bandwidth allocated by congestion control can increase energy consumption by as high as 16%. We extrapolate these results to projected savings on the order of $10 million/year for large data centers.

## CCS CONCEPTS

• **Networks** → **Network economics**; • **Hardware** → **Impact on the environment**; *Enterprise level and data centers power issues*;

## KEYWORDS

Congestion Control, Green Networks, Fairness

**ACM Reference Format:**
Serhat Arslan, Sundararajan Renganathan, and Bruce Spang. 2023. Green With Envy: Unfair Congestion Control Algorithms Can Be More Energy Efficient. In *The 22nd ACM Workshop on Hot Topics in Networks (HotNets '23), November 28–29, 2023, Cambridge, MA, USA*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3626111.3628200

*HotNets '23, November 28–29, 2023, Cambridge, MA, USA*
© 2023 Copyright held by the owner/author(s). Publication rights licensed to the Association for Computing Machinery.
ACM ISBN 979-8-4007-0415-4/23/11…$15.00
https://doi.org/10.1145/3626111.3628200

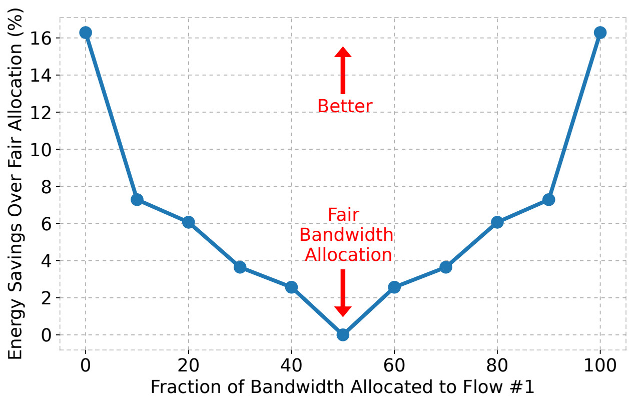

**Figure 1: Increasing throughput imbalance for two competing TCP flows can reduce energy usage.**

## 1 INTRODUCTION

The Internet consumes 400-800 Terawatt-Hours of electricity each year [20, 42]. Data centers alone account for approximately 3% of the global electricity usage [28], a figure projected to rise significantly in the future [5, 29]. Therefore, any improvement in the energy efficiency of communication networks will be beneficial—both environmentally, and financially for network owners.

Our community has done extensive research on congestion control algorithms (CCAs) [1, 3, 4, 7, 12–15, 22, 24, 26, 27, 35, 36, 39, 43, 49, 61]. These algorithms decide when to send traffic into the network, focusing on the optimal utilization of resources, avoiding packet loss, and ensuring fair allocation of bandwidth among competing flows. However, there is relatively little research on the energy impact of different CCAs [56]. If such algorithms impact the energy usage of end-hosts or networking equipment, developers can exploit this to reduce the energy footprint—and cost—of the infrastructure.

In this paper, we explore the energy consequences of congestion control algorithms. We measure their energy usage in a lab setting and find that different CC behavior can have a notable impact on end-host energy usage.

Our main finding is that **increasing throughput unfairness can improve energy efficiency**. To demonstrate this, we ran an experiment where two flows share a 10 Gbps bottleneck link, each transferring 10 Gbit of data. We limited the throughput of one flow, allowing the remaining flow to use the rest of the link. Then, we measured the total energy usage during the experiment—from when the experiment began until both flows successfully completed.

Figure 1 shows the results. Surprisingly, the way most CCAs allocate bandwidth (the TCP fair share allocation where both flows get 5 Gbps) is *the least energy efficient*. As unfairness increases, energy efficiency improves. The most energy-efficient bandwidth allocation is the least fair, the *full speed, then idle approach* where one flow sends all its data at line rate while the other idles, and then idles while the other flow sends at line rate. In this experiment, this approach uses 16% less energy than the fair approach. In §4.1, we show that the TCP fair share is the *worst* allocation for energy efficiency.

We will go into detail about this result and how unfairness improves energy efficiency in §4.1. The intuition is that marginal energy usage is a decreasing function of throughput. As throughput increases, overall energy usage increases but end-hosts become more efficient—the energy required for each additional Gbps of throughput decreases. Sending at line rate allows the flows to finish quickly and efficiently, and then idle to reduce energy usage. Sending at the TCP fair share requires sustained, higher energy usage.

In §4, we also discuss additional findings on how end-host networking choices can impact energy usage at the end-hosts. We show that increasing MTU, reducing flow completion time (FCT), and changing CCAs can all reduce energy usage. These strategies are noteworthy since the digital infrastructure accounts for 3-4% of the global energy usage [28].

Our results suggest that to optimize energy usage, we as a community should rethink our current approach to congestion control. In §5 we go into the consequences of our results on CC design, and more generally on other areas of networking research including load balancing. By making the case for a more comprehensive view of network design and operation, we hope to inspire further research to push the boundary of energy efficiency in communication networks.

## 2 RELATED WORK

CC is one of the most important mechanisms for the efficient utilization of network resources. Modern algorithms report almost full link utilization with minimal queuing [1, 7, 14, 36, 39, 61]. The primary performance metrics used for such algorithms are flow completion time [19], tail queuing [18], and fairness [34]. Further, CCAs are evaluated for the value of the information they use [8, 62], their effect on buffer sizing [52], and friendliness to different algorithms [55]. Yet, none of these works consider energy consumption as a factor when designing or evaluating CCAs.

The CC community for wide area networks has studied Multi-Path TCP (MPTCP) to better utilize all the available paths between the source and the destination [23]. This opened up discussions for how to choose among different available paths and energy was considered as one of the factors [37]. Observations made in [60] suggested that CPU's energy consumption for the transport protocol is directly proportional to average throughput, and path delay. With the same spirit, [59] recommended eliminating link sharing between sub-flows to minimize CPU consumption for the same network resource to save energy in data centers. Our work confirms these insights and claims that similar savings can also be obtained between independent flows that share the same bottleneck.

Choosing among available paths for the sake of minimal energy consumption has also been considered for mobile networks where energy is a scarce resource. In a congested wireless link, retransmissions or control signaling can consume a significant fraction of the energy for communication. This is minimized with routing or CCAs that choose paths or sending times based on the energy potential of the link, packet loss rate, and congestion level [2, 40, 57].

One other approach for reducing energy consumption in networks is to address the energy footprint of switches. Prior work suggests that the load on an active switch does not affect the power draw [21, 32]. However, reducing the line rate or turning off the links completely during idle periods can reduce energy consumption [44, 46]. Our work focuses on the energy footprint of end-hosts' transport layer, which is complementary to reducing the energy footprint of switches.

The energy footprint of end-hosts has been studied for cloud networks. Carefully placing VMs and balancing traffic flows was shown to save energy up to 50% [54]. Similarly, [33] proposed to consolidate the VMs of a large group of users in the cloud to save energy instead of dedicating a physical machine to each user. Our work is complementary and looks at how CC can further reduce the energy footprint.

[21] shared data about energy usage of data centers, and discussed reducing consumption at end-hosts by CPU voltage scaling and reducing idle energy consumption. The same researchers also showed that power is a strictly concave, increasing function of CPU utilization in data centers [10]. An argument similar to our results, suggesting that it is more energy-efficient to serve the load from a few, heavily loaded servers rather than fairly balancing the load across all available servers. This approach was explored in prior work for load balancing [11, 31, 41, 53].

Our findings on energy savings with unfair traffic allocation have parallels with recent findings on improving performance in machine learning clusters. Rajasekaran et al. [48] make the observation that artificially creating unfair bandwidth allocation to different multi-server ML tasks can prevent synchronization between the tasks, facilitate better utilization of the network resources for the entire training, and reduce the training times of the models. Not only can unfairness improve energy savings, it can also reduce flow completion times.

## 3 EXPERIMENT SETUP

We use measurements from our physical testbed to understand the energy consumption of networking and CC. Our testbed consists of an Intel Tofino switch [17] and two servers, each with 32 GB of RAM and a pair of 2.4Ghz Intel Xeon E5-2630 v3 CPUs with 16 hyper-threaded cores each. The servers run Linux 5.10.0 and are connected to the network with Intel 82599ES 10Gb/s NICs. The sender server is connected to the switch with 2×10Gb/s links where the interfaces are bonded and packets are sent round-robin among the two. This ensures that the bottleneck for all the experiments is the switch rather than the sender's NIC. We use an MTU of 9000 bytes (unless stated otherwise) to achieve the full 10 Gb/s line rate.

The traffic for our experiments is generated via `iperf3` with the CCAs included in the Linux kernel. We used TCP Reno [12], CUBIC [49], DCTCP [3], BBR (v1) [14], Vegas [13], Scalable [35], Westwood [24], and Highspeed TCP [22] as well as Google's alpha release of BBR2 [15] in our experiments. In addition, we have created a new kernel module that replaces any CC mechanism with a large, constant `cwnd` value. We use this module as the `baseline` to compare the energy consumption of CC-only computations.

We measure the energy consumption of the servers using Intel's Running Average Power Limit (RAPL) interface [47] which accurately estimates the consumption in the CPUs via software models [50]. The models maintain counters to keep track of the cumulative energy used by the CPUs. For each scenario, we read the energy counter for each CPU before and after the experiment. The difference between the successive counter reads gives us the energy used by the scenario for that CPU. This enables automatic measurement of the energy consumption for `iperf3` traffic. We repeat each scenario 10 times and report standard deviations of our results.[1]

## 4 RESULTS

We investigate the energy usage of end-hosts' network stack, with an emphasis on congestion control. Our main finding is presented in §4.1 which is generalized for loaded servers in §4.2, leading to a prospective strategy for flow scheduling and CC for greener energy footprint. We further discuss the energy consumption for transmiting a certain amount of bytes with different CCAs and MTU sizes in §4.3 and §4.4, respectively. Finally, we list major performance metrics for CC that correlate with the energy consumption in §4.5.

### 4.1 Unfair Allocation Reduces Energy Usage

In this section, we show that a TCP-fair allocation is the least energy efficient bandwidth allocation under mild assumptions about how throughput affects energy usage. We describe the

[1]Our source code for running and analyzing these experiments can be found at https://github.com/brucespang/power.

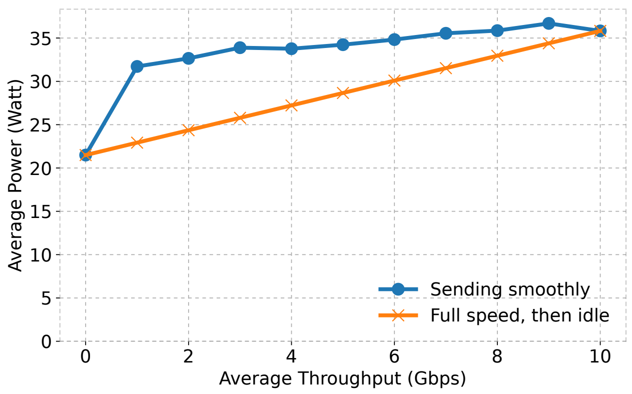

**Figure 2: Rate of energy consumption for a CUBIC sender while sending at different throughputs.**

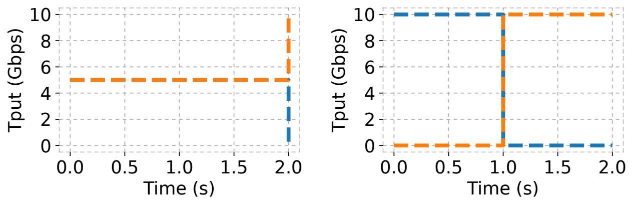

**Figure 3: Sending at line rate (on the right) allows each flow to idle while the other sends leading to lower energy usage than maintaining fair throughput (on the left).**

results of a lab experiment where two CUBIC flows share a 10 Gb/s bottleneck link, each transferring 10 Gbit of data. We limit the throughput of one flow, allowing the other flow to use the remaining bandwidth, and measure the total energy usage until both flows successfully complete.

Figure 1 shows the results. Energy efficiency improves as unfairness increases. The most efficient bandwidth allocation is the least fair, the *"full speed, then idle"* approach: one flow sends all its data at line rate while the other idles, and idles while the other flow sends at line rate. This consumes 16% less energy than both flows sending at 5 Gbps until completion.

Looking closer, Figure 2 shows the power usage (energy used per second) when CUBIC is limited to different average throughputs. A throughput of zero corresponds to the server idling. The blue line corresponds to sending smoothly at a certain throughput over the duration of the experiment.

Note that in Figure 2, power usage is a strictly concave function of throughput. Sending with 5 additional Gb/s increases power usage by 60% (12.7 Watts) when the server is idling, but only increases it by 5% (1.6 Watts) when the server is already sending at 5 Gb/s. In this case, increasing unfairness reduces power usage. One possible explanation for concavity is that increasing throughput linearly increases CPU utilization in our experiments, and CPU utilization has been observed to be a concave function of power usage [21].

As an example of how Figure 2 implies that unfairness improves energy efficiency, we calculate the power usage in two

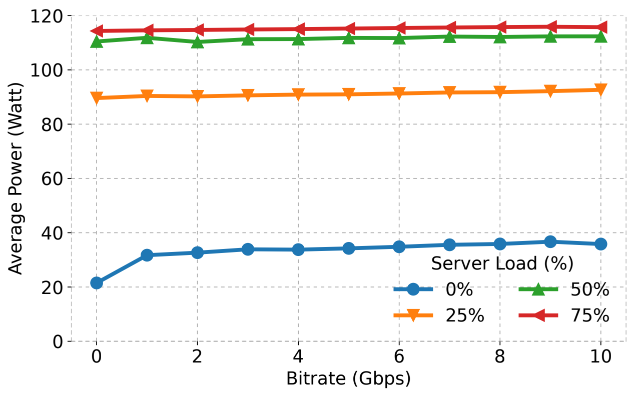

**Figure 4: Rate of energy consumption for a CUBIC sender with different amounts of server loads in the background.**

different settings: a fair scenario when both flows send equally at 5 Gb/s and the "*full speed, then idle*" approach. Throughput over time for these flows is depicted in Figure 3. In all cases, all flows have the same 5 Gbps average throughput.

In the fair scenario (left), both of the flows would send at 5 Gbps and take 2 seconds to finish. Based on Figure 2, when both flows send at 5 Gbps, they use 34.23 Watts for two seconds, for a total of 137 Watts. With the "full speed, then idle" approach (right), flows switch between sending all their data at line rate and idling. Each flow sends at 10 Gbps and uses 35.82 Watts for 1 second. The instantaneous power usage is 5% higher than when they both send at 5 Gbps. However, during their idle period, each flow consumes only 21.49 Watts. Over the entire experiment, the flows consume a total of 114.63 Watts, 16% less than in the fair scenario.

In general, whenever marginal power usage is a decreasing function of throughput, fairness is the *least energy efficient* thing to do. We formalize this with the following theorem:

THEOREM 1. *Let $x \in \mathbb{R}_{>\not{r}}{}^n$ be the throughputs of n flows that share a link of capacity C. Let $P(x) = \sum_{i=1}^{n} p(x_i)$ be the power usage given the throughputs x. Let $x^* = \{C/n, \ldots, C/n\}$, and let y be some other set of throughputs where $\sum_{i=0}^{n} y_i = C$. If $p(x)$ is a strictly concave function, then $P(x^*) > P(y)$.*

PROOF. Since $\sum_{i=0}^{n} y_i = C$,

$$P(x^*) = np(C/n) = np(\frac{y_1}{n} + \ldots + \frac{y_n}{n}). \qquad (1)$$

By strict concavity,

$$p(\frac{1}{n}y_1 + \ldots + \frac{1}{n}y_n) > \frac{1}{n}p(y_1) + \ldots + \frac{1}{n}p(y_n).$$

Substituting into (1) yields the desired result as

$$P(x^*) > p(y_1) + \ldots + p(y_n) = P(y).$$

□

We depict this argument visually in Figure 2. A throughput of $x$ can be achieved either by sending at $x$ smoothly or by sending at full speed and then idling. If we send at full speed and then idle, the power usage is a linear combination of the full speed and the idle power – on the orange tangent line in Figure 2. Since power usage is a strictly concave function, this is strictly less power than sending smoothly.

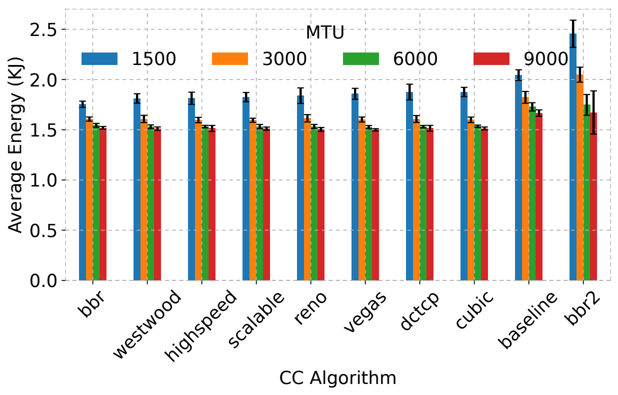

**Figure 5: Average energy consumption of the CCAs to transmit 50 GB of data**

## 4.2 Unfairness is Green For Loaded Hosts Too

To mimic more realistic scenarios, we also explore energy consumption with different bandwidths while running background computations on the end-hosts. For this, we use the `stress` tool in Linux and generate load on a certain number of cores at the end-host in addition to the CUBIC traffic. Figure 4 shows the energy measurements for various compute loads with respect to the communication bitrate.

As expected, reducing the networking-related energy consumption reduces overall consumption less when the server is already loaded with compute compared to when it is idle. Nonetheless, the "full speed, then idle" approach can still save 1% when the server load is 25%, or 0.17% when the server load is 75%. This is a small percentage but can lead to significant energy savings at scale. The energy to run a typical data center rack is on the order of $10k/year [51]. With around 100k racks in a typical data center [38], a 1% improvement corresponds to a cost savings of on the order of $10 million/year.

## 4.3 Energy Consumption of CC Algorithms Differ Notably

In addition to flow scheduling strategies, the use of different CCAs also changes energy consumption. To demonstrate this, we transmit 50 GB of data while using different algorithms at the sender and measure the energy consumption. Since the same amount of data is sent, the measurements capture the same amount of serialization-related energy consumption. Then, the difference in the measurements is limited to the CC-related computations and the inflicted queuing operations

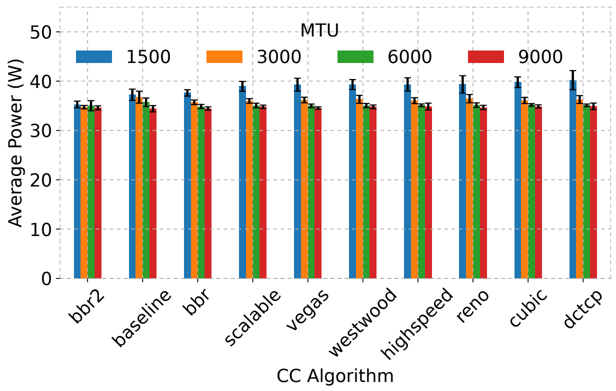

**Figure 6: Rate of energy consumption for the CCAs to transmit 50 GB of data.**

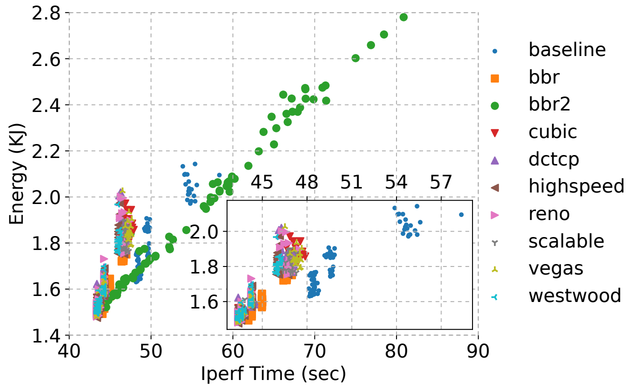

**Figure 7: Energy consumption vs flow completion time for different CCAs transmitting 50 GB of data**

on the machine which are the direct consequences of CCA design. Figure 5 shows the results of this experiment. The error bars on the figure represent the standard deviation of the measurements among different runs of the same scenario.

We note that all the algorithms, except BBR2, consume 8.2% to 14.2% less energy than our custom `baseline` which does not perform any CC computations. Instead, it uses a constantly large `cwnd` value while running the same logic for other TCP mechanisms, *i.e.*, retransmission timeouts, selective acknowledgments, and loss recovery.[2] Although it doesn't run any logic to recompute `cwnd` value frequently, its large `cwnd` value makes the sender bursty which causes queuing at the network as well as the sender host resulting in more frequent memory accesses and packet loss. This confirms that effective CC can reduce the energy overhead of utilizing network resources Hence, energy consequences of algorithms should be evaluated when designing them.

We also observe that the energy footprint differs by 40% between the versions of BBR. Although we have not been able to identify the exact reason for this significant difference yet, we acknowledge that the BBR2 version in our testbed is the alpha release of the algorithm which might be lacking efficient implementation or prone to undiscovered bugs. Nonetheless, it remains a strong signal that implementation maturity can play an important role in the energy footprint of algorithms and proper investment in this can help obtain savings.

We also investigate the rate of energy consumption, *i.e.*, power, for the same set of CCAs. Figure 6 shows the average amount of energy consumed per second for each algorithm while transmitting 50 GB of data. The power difference between CCAs is about 14% which is significant considering how much it would correspond to in large scale.

Note that both Figure 5 and Figure 6 list the CCAs with an increasing order of energy usage for 1500 Bytes of MTU scenarios. The order of algorithms change drastically between the two figures. We compute the correlation between total energy consumption vs power as $-0.8$. This confirms our results in §4.1 that hosts may spend less energy per unit of time, but take longer to complete and end up spending more energy in total. As the senders stay active for longer times, they pay the price for the overhead of keeping the server active. More discussion about the effect of flow completion times on energy consumption is provided in §4.5.

## 4.4 Increasing MTU Saves Energy

We also ran experiments to measure the impact of different MTU sizes on energy usage as an example for how network configuration may affect the energy footprint. The standard MTU on the Internet is 1500 Bytes, but data center operators often use larger MTU sizes to reduce the overhead of packet processing and achieve higher line rates [25]. Our results confirm that larger MTUs can also reduce energy consumption which is a known rule of thumb among switch designers.

With a single TCP connection transferring 50 GB of data, we measured the energy consumption with standard 1500 byte MTUs, and compared to MTUs of 3, 6, and 9 KBytes. Figure 5 presents the average energy spent throughout the experiment for different configurations. As the MTU increases, the number of packets generated to transmit the same amount of data decreases, reducing the total overhead of packet processing on the host. As a consequence, the total amount of energy spent also decreases. In our results, increasing MTU from 1500 bytes to 9KB can decrease energy consumption by 13.4% to 31.9%, depending on the CCA.

## 4.5 Reducing Flow Completion Time Improves Energy Consumption

In our experiments, energy consumption is strongly correlated with the flow completion time: the total time taken by

[2]This mechanism would create a congestion collapse described in [30]. Therefore we never use this module when there are multiple flows in the network.

`iperf3` to deliver 50 GB of data. Figure 7 shows the relationship between energy consumption and the time it takes to send the data for different CCAs. Note that the figure also includes the measurements with different MTU sizes, hence there are two major clusters of energy measurements also shown within the axis inset. The cluster on the bottom left corner of the inset is the group of measurements with large MTU sizes whereas the cluster on the top right corner of the inset is the measurements with an MTU size of 1500 Bytes.

One reason for the different FCT among CCAs is their ability to maintain high throughput by avoiding retransmissions. Figure 8 shows the relationship between the number of retransmissions and the energy consumption for the same experiments. The overall correlation between the energy consumption and the retransmission is calculated as 0.47 excluding the highly variable BBR2 measurements. The absence of congestion control (`baseline`) naturally induces a higher rate of retransmissions and ends up consuming a larger amount of energy on average for all the MTU sizes tested. These results indicate that designing a CCA that is capable of finishing flows faster while achieving lower rates of packet loss is required also for environment-friendliness in addition to the high-performance requirements of data center workloads.

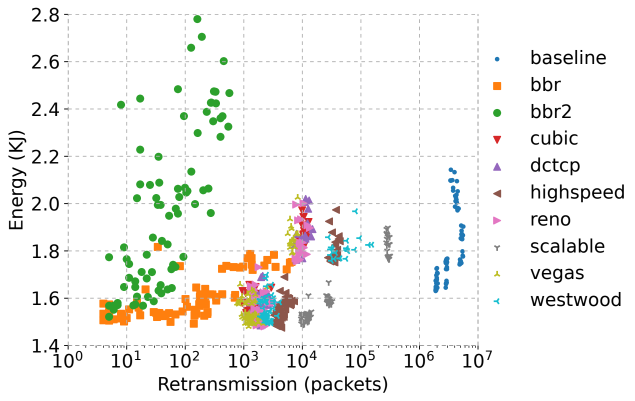

**Figure 8: Energy consumption vs retransmissions for different CCAs transmitting 50GB of data**

## 5 FUTURE WORK & CONCLUSION

Congestion control algorithms focus on efficiently using network resources, but their energy footprint has been largely overlooked. In this work, we have begun to build an understanding of how networking algorithms can improve the energy usage of wireline networks. We believe that sharing these results with our community can create a stream of research that is actively working towards building greener data centers in the future. To conclude, we discuss these future research directions.

Our main result is that when marginal power usage is a decreasing function of throughput, it is the most energy efficient to be unfair. Our experiments confirm this in simple lab settings, and it is not surprising given optimizations like TSO which help modern transport layers run at line rate. Investigating if this holds at scale, with hardware offloading [6], and with the sorts of workloads used in production data centers is needed as future work, including multiplexing multiple flows at the same sender, and incast.

Higher energy efficiency by unfairness also has significant consequences for how we design CCAs. Today, most CCAs are specifically designed for fairness [1, 3, 7, 36, 39, 61]. Our results suggest that to improve energy efficiency, CCAs should aim to send as fast as possible for minimal completion time. One intriguing approach would be to measure the energy usage of existing transport protocols that approximate the Shortest Remaining Processing Time first (SRPT) scheduling [4, 9, 27, 43].

Although we show that using different CCAs can impact energy consumption, our results in §4.3 does not necessarily expose the underlying reason for these differences. We expect such differences to stem from unique mechanisms used for each algorithm such as maintained flow state, packet pacing, `cwnd` calculation arithmetic, and so on. We plan to investigate the energy consequences of such mechanisms in future work. This work would provide a clear guideline for how to design green, high-performance CCAs.

We also acknowledge that our work is based on a relatively small subset of CCAs in the literature. It is particularly intriguing for us to evaluate production algorithms of large data centers, *i.e.,* Swift [36], DCQCN [61], and HPCC [39]. Unfortunately, not all algorithms have publicly available commercial implementations, and evaluating them based on personal implementations bears the risk of measuring inefficient implementations as well as bugs. Therefore, we invite the community to build a benchmark for a standardized evaluation of such algorithms. This would also enable the comparison of new CC designs to existing work.

Finally, prior work suggests that utilization does not significantly affect the energy consumption of switches and routers [21, 32]. In the meantime, [45] has suggested that networking equipment *should* be built to reduce power usage when the load is reduced. If a data center contained such equipment, our results imply that there could be significant power savings by increasing load imbalance across data center links. This would have consequences for routing algorithms and load balancing algorithms (see [16, 58] for surveys). The design and development of such network equipment remains as an unexplored avenue for future data centers.

## ACKNOWLEDGMENTS

We thank Nick McKeown for his invaluable feedback. This work was supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-20-C-0107 and FA8650-18-2-7865.

## REFERENCES

[1] Vamsi Addanki, Oliver Michel, and Stefan Schmid. 2022. PowerTCP: Pushing the Performance Limits of Datacenter Networks. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*. USENIX Association, Renton, WA, 51–70. https://www.usenix.org/conference/nsdi22/presentation/addanki

[2] Omar Ahmed, Fuji Ren, Ammar Hawbani, and Yaser Al-Sharabi. 2020. Energy Optimized Congestion Control-Based Temperature Aware Routing Algorithm for Software Defined Wireless Body Area Networks. *IEEE Access* 8 (2020), 41085–41099. https://doi.org/10.1109/ACCESS.2020.2976819

[3] Mohammad Alizadeh, Albert Greenberg, David A. Maltz, Jitendra Padhye, Parveen Patel, Balaji Prabhakar, Sudipta Sengupta, and Murari Sridharan. 2010. Data Center TCP (DCTCP). *SIGCOMM Comput. Commun. Rev.* 40, 4 (Aug. 2010), 63–74. https://doi.org/10.1145/1851275.1851192

[4] Mohammad Alizadeh, Shuang Yang, Milad Sharif, Sachin Katti, Nick McKeown, Balaji Prabhakar, and Scott Shenker. 2013. PFabric: Minimal near-Optimal Datacenter Transport. *SIGCOMM Comput. Commun. Rev.* 43, 4 (Aug. 2013), 435–446. https://doi.org/10.1145/2534169.2486031

[5] Anders SG Andrae and Tomas Edler. 2015. On Global Electricity Usage of Communication Technology: Trends to 2030. *Challenges* 6, 1 (2015), 117–157.

[6] Serhat Arslan, Stephen Ibanez, Alex Mallery, Changhoon Kim, and Nick McKeown. 2021. NanoTransport: A Low-Latency, Programmable Transport Layer for NICs. In *Proceedings of the ACM SIGCOMM Symposium on SDN Research (SOSR) (SOSR '21)*. Association for Computing Machinery, New York, NY, USA, 13–26. https://doi.org/10.1145/3482898.3483365

[7] Serhat Arslan, Yuliang Li, Gautam Kumar, and Nandita Dukkipati. 2023. Bolt: Sub-RTT Congestion Control for Ultra-Low Latency. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*. USENIX Association, Boston, MA, 219–236. https://www.usenix.org/conference/nsdi23/presentation/arslan

[8] Serhat Arslan and Nick McKeown. 2019. Switches Know the Exact Amount of Congestion. In *Proceedings of the 2019 Workshop on Buffer Sizing (BS '19)*. Association for Computing Machinery, New York, NY, USA, Article 10, 6 pages. https://doi.org/10.1145/3375235.3375245

[9] Wei Bai, Li Chen, Kai Chen, Dongsu Han, Chen Tian, and Hao Wang. 2015. Information-Agnostic Flow Scheduling for Commodity Data Centers. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. 455–468.

[10] Luiz André Barroso and Urs Hölzle. 2007. The Case for Energy-Proportional Computing. *Computer* 40, 12 (2007), 33–37. https://doi.org/10.1109/MC.2007.443

[11] R. Bianchini and R. Rajamony. 2004. Power and Energy Management for Server Systems. *Computer* 37, 11 (Nov. 2004), 68–76. https://doi.org/10.1109/MC.2004.217

[12] Ethan Blanton, Dr. Vern Paxson, and Mark Allman. 2009. TCP Congestion Control. RFC 5681. (Sept. 2009). https://doi.org/10.17487/RFC5681

[13] Lawrence S. Brakmo, Sean W. O'Malley, and Larry L. Peterson. 1994. TCP Vegas: New Techniques for Congestion Detection and Avoidance. In *Proceedings of the Conference on Communications Architectures, Protocols and Applications (SIGCOMM '94)*. Association for Computing Machinery, New York, NY, USA, 24–35. https://doi.org/10.1145/190314.190317

[14] Neal Cardwell, Yuchung Cheng, C. Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson. 2017. BBR: Congestion-Based Congestion Control. *Commun. ACM* 60, 2 (jan 2017), 58–66. https://doi.org/10.1145/3009824

[15] Neal Cardwell, Yuchung Cheng, Soheil Hassas Yeganeh, Ian Swett, Victor Vasiliev, Priyaranjan Jha, Yousuk Seung, Matt Mathis, and Van Jacobson. 2019. BBR v2: A Model-Based Congestion Control. (March 2019). https://www.ietf.org/proceedings/104/slides/slides-104-iccrg-an-update-on-bbr-00

[16] Kai Chen, Chengchen Hu, Xin Zhang, Kai Zheng, Yan Chen, and Athanasios V. Vasilakos. 2011. Survey on Routing in Data Centers: Insights and Future Directions. *IEEE Network* 25, 4 (July 2011), 6–10. https://doi.org/10.1109/MNET.2011.5958002

[17] Intel Corporation. 2023. Tofino: P4-programmable Ethernet switch ASIC that delivers better performance at lower power. (May 2023). https://www.intel.com/content/www/us/en/products/network-io/programmable-ethernet-switch/tofino-series.html

[18] Jeffrey Dean and Luiz André Barroso. 2013. The Tail at Scale. *Commun. ACM* 56 (2013), 74–80. http://cacm.acm.org/magazines/2013/2/160173-the-tail-at-scale/fulltext

[19] Nandita Dukkipati and Nick McKeown. 2006. Why Flow-Completion Time is the Right Metric for Congestion Control. *SIGCOMM Comput. Commun. Rev.* 36, 1 (jan 2006), 59–62. https://doi.org/10.1145/1111322.1111336

[20] Thunder Said Energy. 2023. What is the energy consumption of the internet? (April 2023). https://thundersaidenergy.com/2023/04/20/what-is-the-energy-consumption-of-the-internet/

[21] Xiaobo Fan, Wolf-Dietrich Weber, and Luiz Andre Barroso. 2007. Power Provisioning for a Warehouse-Sized Computer. In *Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA '07)*. Association for Computing Machinery, New York, NY, USA, 13–23. https://doi.org/10.1145/1250662.1250665

[22] Sally Floyd. 2003. HighSpeed TCP for Large Congestion Windows. RFC 3649. (Dec. 2003). https://doi.org/10.17487/RFC3649

[23] Alan Ford, Costin Raiciu, Mark J. Handley, Olivier Bonaventure, and Christoph Paasch. 2020. TCP Extensions for Multipath Operation with Multiple Addresses. RFC 8684. (March 2020). https://doi.org/10.17487/RFC8684

[24] M. Gerla, M.Y. Sanadidi, Ren Wang, A. Zanella, C. Casetti, and S. Mascolo. 2001. TCP Westwood: congestion window control using bandwidth estimation. In *GLOBECOM'01. IEEE Global Telecommunications Conference (Cat. No.01CH37270)*, Vol. 3. 1698–1702 vol.3. https://doi.org/10.1109/GLOCOM.2001.965869

[25] Dan Gibson, Hema Hariharan, Eric Lance, Moray McLaren, Behnam Montazeri, Arjun Singh, Stephen Wang, Hassan M. G. Wassel, Zhehua Wu, Sunghwan Yoo, Raghuraman Balasubramanian, Prashant Chandra, Michael Cutforth, Peter Cuy, David Decotigny, Rakesh Gautam, Alex Iriza, Milo M. K. Martin, Rick Roy, Zuowei Shen, Ming Tan, Ye Tang, Monica Wong-Chan, Joe Zbiciak, and Amin Vahdat. 2022. Aquila: A unified, low-latency fabric for datacenter networks. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*. USENIX Association, Renton, WA, 1249–1266. https://www.usenix.org/conference/nsdi22/presentation/gibson

[26] Mark Handley, Costin Raiciu, Alexandru Agache, Andrei Voinescu, Andrew W. Moore, Gianni Antichi, and Marcin Wójcik. 2017. Re-Architecting Datacenter Networks and Stacks for Low Latency and High Performance. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication (SIGCOMM '17)*. Association for Computing Machinery, New York, NY, USA, 29–42. https://doi.org/10.1145/3098822.3098825

[27] Shuihai Hu, Wei Bai, Gaoxiong Zeng, Zilong Wang, Baochen Qiao, Kai Chen, Kun Tan, and Yi Wang. 2020. Aeolus: A Building Block

for Proactive Transport in Datacenters. In *Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM '20)*. Association for Computing Machinery, New York, NY, USA, 422–434. https://doi.org/10.1145/3387514.3405878

[28] AFL Hyperscale. 2023. What Makes Hyperscale, Hyperscale? (March 2023). https://www.aflhyperscale.com/articles/what-makes-hyperscale-hyperscale/

[29] IEA. Fetched June 25th, 2023. Data Centres and Data Transmission Networks. (Fetched June 25th, 2023). https://www.iea.org/reports/data-centres-and-data-transmission-networks.

[30] Van Jacobson. 1988. Congestion Avoidance and Control. In *Symposium Proceedings on Communications Architectures and Protocols (SIGCOMM '88)*. Association for Computing Machinery, New York, NY, USA, 314–329. https://doi.org/10.1145/52324.52356

[31] Einollah Jafarnejad Ghomi, Amir Masoud Rahmani, and Nooruldeen Nasih Qader. 2017. Load-Balancing Algorithms in Cloud Computing: A Survey. *Journal of Network and Computer Applications* 88 (June 2017), 50–71. https://doi.org/10.1016/j.jnca.2017.04.007

[32] Maria Kazandjieva, Brandon Heller, Omprakash Gnawali, Philip Levis, and Christos Kozyrakis. 2013. Measuring and Analyzing the Energy Use of Enterprise Computing Systems. *Sustainable Computing: Informatics and Systems* 3, 3 (2013), 218–229. https://doi.org/10.1016/j.suscom.2013.01.009

[33] Maria Kazandjieva, Chinmayee Shah, Ewen Cheslack-Postava, Behram Mistree, and Philip Levis. 2014. System architecture support for green enterprise computing. In *International Green Computing Conference*. 1–10. https://doi.org/10.1109/IGCC.2014.7039143

[34] Frank Kelly. 2003. Fairness and Stability of End-to-End Congestion Control*. *European Journal of Control* 9, 2 (2003), 159–176. https://doi.org/10.3166/ejc.9.159-176

[35] Tom Kelly. 2003. Scalable TCP: Improving Performance in Highspeed Wide Area Networks. *SIGCOMM Comput. Commun. Rev.* 33, 2 (apr 2003), 83–91. https://doi.org/10.1145/956981.956989

[36] Gautam Kumar, Nandita Dukkipati, Keon Jang, Hassan M. G. Wassel, Xian Wu, Behnam Montazeri, Yaogong Wang, Kevin Springborn, Christopher Alfeld, Michael Ryan, David Wetherall, and Amin Vahdat. 2020. Swift: Delay is Simple and Effective for Congestion Control in the Datacenter. In *Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM '20)*. Association for Computing Machinery, New York, NY, USA, 514–528. https://doi.org/10.1145/3387514.3406591

[37] Tuan Anh Le, Choong Seon Hong, Md. Abdur Razzaque, Sungwon Lee, and Heeyoung Jung. 2012. ecMTCP: An Energy-Aware Congestion Control Algorithm for Multipath TCP. *IEEE Communications Letters* 16, 2 (2012), 275–277. https://doi.org/10.1109/LCOMM.2011.120211.111818

[38] Julia Leonard. 2021. Racks In The Google Data Center? (Sept. 2021). https://www.akibia.com/racks-in-the-google-data-center/

[39] Yuliang Li, Rui Miao, Hongqiang Harry Liu, Yan Zhuang, Fei Feng, Lingbo Tang, Zheng Cao, Ming Zhang, Frank Kelly, Mohammad Alizadeh, and Minlan Yu. 2019. HPCC: High Precision Congestion Control. In *Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM '19)*. Association for Computing Machinery, New York, NY, USA, 44–58. https://doi.org/10.1145/3341302.3342085

[40] Uzma Majeed, Aqdas Naveed Malik, Nasim Abbas, and Waseem Abbass. 2022. An Energy-Efficient Distributed Congestion Control Protocol for Wireless Multimedia Sensor Networks. *Electronics* 11, 20 (2022). https://doi.org/10.3390/electronics11203265

[41] Vimal Mathew, Ramesh K. Sitaraman, and Prashant Shenoy. 2011. Energy-Aware Load Balancing in Content Delivery Networks. (Sept. 2011). arXiv:cs/1109.5641 http://arxiv.org/abs/1109.5641

[42] Christopher McFadden. 2023. Internet energy usage: How the life-changing network has a hidden cost. (Jan. 2023). https://interestingengineering.com/innovation/whats-the-energy-cost-internet#

[43] Behnam Montazeri, Yilong Li, Mohammad Alizadeh, and John Ousterhout. 2018. Homa: A Receiver-Driven Low-Latency Transport Protocol Using Network Priorities. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication (SIGCOMM '18)*. Association for Computing Machinery, New York, NY, USA, 221–235. https://doi.org/10.1145/3230543.3230564

[44] Sergiu Nedevschi, Lucian Popa, Gianluca Iannaccone, Sylvia Ratnasamy, and David Wetherall. 2008. Reducing Network Energy Consumption via Sleeping and Rate-Adaptation. In *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation (NSDI'08)*. USENIX Association, USA, 323–336.

[45] Sergiu Nedevschi, Lucian Popa, Gianluca Iannaccone, Sylvia Ratnasamy, and David Wetherall. 2008. Reducing Network Energy Consumption via Sleeping and Rate-Adaptation. In *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation (NSDI'08)*. USENIX Association, USA, 323–336.

[46] Carla Panarello, Alfio Lombardo, Giovanni Schembra, Michela Meo, Marco Mellia, and Marco Ajmone Marsan. 2013. Power management and TCP congestion control: Friends or foes?. In *2013 22nd ITC Specialist Seminar on Energy Efficient and Green Networking (SSEEGN)*. 43–49. https://doi.org/10.1109/SSEEGN.2013.6705401

[47] Srinivas Pandruvada. 2014. RUNNING AVERAGE POWER LIMIT – RAPL. (June 2014). https://01.org/blogs/2014/running-average-power-limit-âĂŞ-rapl

[48] Sudarsanan Rajasekaran, Manya Ghobadi, Gautam Kumar, and Aditya Akella. 2022. Congestion Control in Machine Learning Clusters. In *Proceedings of the 21st ACM Workshop on Hot Topics in Networks (HotNets '22)*. Association for Computing Machinery, New York, NY, USA, 235–242. https://doi.org/10.1145/3563766.3564115

[49] Injong Rhee, Lisong Xu, Sangtae Ha, Alexander Zimmermann, Lars Eggert, and Richard Scheffenegger. 2018. CUBIC for Fast Long-Distance Networks. RFC 8312. (Feb. 2018). https://doi.org/10.17487/RFC8312

[50] Efraim Rotem, Alon Naveh, Avinash Ananthakrishnan, Eliezer Weissmann, and Doron Rajwan. 2012. Power-Management Architecture of the Intel Microarchitecture Code-Named Sandy Bridge. *IEEE Micro* 32, 2 (2012), 20–27. https://doi.org/10.1109/MM.2012.12

[51] Mike Schmitt. 2021. How Much Does it Cost to Power One Rack in a Data Center? (July 2021). https://www.nlyte.com/blog/how-much-does-it-cost-to-power-one-rack-in-a-data-center/

[52] Bruce Spang, Serhat Arslan, and Nick McKeown. 2021. Updating the theory of buffer sizing. *Performance Evaluation* 151 (2021), 102232. https://doi.org/10.1016/j.peva.2021.102232

[53] Nedeljko Vasić, Martin Barisits, Vincent Salzgeber, and Dejan Kostic. 2009. Making Cluster Applications Energy-Aware. In *Proceedings of the 1st Workshop on Automated Control for Datacenters and Clouds*. ACM, Barcelona Spain, 37–42. https://doi.org/10.1145/1555271.1555281

[54] Lin Wang, Fa Zhang, Jordi Arjona Aroca, Athanasios V. Vasilakos, Kai Zheng, Chenying Hou, Dan Li, and Zhiyong Liu. 2014. GreenDCN: A General Framework for Achieving Energy Efficiency in Data Center Networks. *IEEE Journal on Selected Areas in Communications* 32, 1 (2014), 4–15. https://doi.org/10.1109/JSAC.2014.140102

[55] Ranysha Ware, Matthew K. Mukerjee, Srinivasan Seshan, and Justine Sherry. 2019. Beyond Jain's Fairness Index: Setting the Bar For The

Deployment of Congestion Control Algorithms. In *Proceedings of the 18th ACM Workshop on Hot Topics in Networks (HotNets '19)*. Association for Computing Machinery, New York, NY, USA, 17–24. https://doi.org/10.1145/3365609.3365855

[56] Michael Welzl. 2022. Not a Trade-Off: On the Wi-Fi Energy Efficiency of Effective Internet Congestion Control. In *2022 17th Wireless On-Demand Network Systems and Services Conference (WONS)*. 1–4. https://doi.org/10.23919/WONS54113.2022.9764413

[57] Saneh Lata Yadav, R. L. Ujjwal, Sushil Kumar, Omprakash Kaiwartya, Manoj Kumar, and Pankaj Kumar Kashyap. 2021. Traffic and Energy Aware Optimization for Congestion Control in Next Generation Wireless Sensor Networks. *Journal of Sensors* 2021 (30 Jun 2021), 5575802. https://doi.org/10.1155/2021/5575802

[58] Jiao Zhang, F. Richard Yu, Shuo Wang, Tao Huang, Zengyi Liu, and Yunjie Liu. 2018. Load Balancing in Data Center Networks: A Survey. *IEEE Communications Surveys & Tutorials* 20, 3 (2018), 2324–2352. https://doi.org/10.1109/COMST.2018.2816042

[59] Jia Zhao, Jiangchuan Liu, Haiyang Wang, Chi Xu, Wei Gong, and Changqiao Xu. 2020. Measurement, Analysis, and Enhancement of Multipath TCP Energy Efficiency for Datacenters. *IEEE/ACM Transactions on Networking* 28, 1 (2020), 57–70. https://doi.org/10.1109/TNET.2019.2950908

[60] Jia Zhao, Jiangchuan Liu, Haiyang Wang, Changqiao Xu, and Hongke Zhang. 2023. Multipath Congestion Control: Measurement, Analysis, and Optimization From the Energy Perspective. *IEEE Transactions on Network Science and Engineering* (2023), 1–12. https://doi.org/10.1109/TNSE.2023.3257034

[61] Yibo Zhu, Haggai Eran, Daniel Firestone, Chuanxiong Guo, Marina Lipshteyn, Yehonatan Liron, Jitendra Padhye, Shachar Raindel, Mohamad Haj Yahia, and Ming Zhang. 2015. Congestion Control for Large-Scale RDMA Deployments. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication (SIGCOMM '15)*. Association for Computing Machinery, New York, NY, USA, 523–536. https://doi.org/10.1145/2785956.2787484

[62] Yibo Zhu, Monia Ghobadi, Vishal Misra, and Jitendra Padhye. 2016. ECN or Delay: Lessons Learnt from Analysis of DCQCN and TIMELY. In *Proceedings of the 12th International on Conference on Emerging Networking EXperiments and Technologies (CoNEXT '16)*. Association for Computing Machinery, New York, NY, USA, 313–327. https://doi.org/10.1145/2999572.2999593