The Internet of tomorrow must sleep more and grow old

Romain Jacob
ETH Zürich

Laurent Vanbever
ETH Zürich

Abstract

Today, the ICT industry has a massive carbon footprint (a few percent of the worldwide emissions) and one of the fastest growth rates. The Internet accounts for a large part of that footprint while being also energy inefficient; *i.e.*, the total energy cost per byte transmitted is very high. Thankfully, there are many ways to improve on the current status; we discuss two relatively unexplored directions in this paper.

Putting network devices to "sleep," *i.e.*, turning them off, is known to be an efficient vector to save energy; we argue that harvesting this potential requires new routing protocols, better suited to devices switching on/off often, and revising the corresponding hardware/software co-design. Moreover, we can reduce the embodied carbon footprint by using networking hardware longer, and we argue that this could even be beneficial for reliability! We sketch our first ideas in these directions and outline practical challenges that we (as a community) need to address to make the Internet more sustainable.

1 Introduction

The growth of the Information and Communication Technology (ICT) industry is staggering in terms of revenue, societal importance—and carbon emissions. ICT was estimated to account for 2% of worldwide CO_2 emissions in 2007 and 4% in 2021 [31]. Hypergiants plan to open around one hundred hyperscale datacenters[1] per year over the next three years [37]—that is, one every three days on average. As the demand will continue to grow, reducing the ICT carbon footprint requires addressing the energy (in)efficiencies of today's networks.

The Internet has become a vital component of our societies: ubiquitous, reliable, but energy inefficient [27]. Making the Internet more sustainable must be a prime goal for the coming decade. While history shows that the Internet does not change as long as "it works" [28], we argue that continuing to waste so much energy (and even more, given the expected growth) is unacceptable; **the Internet has to change**. Not pushing for research in that direction feels like a criminal omission.

No single tweak can magically reduce the Internet's carbon footprint. We identify two classes of objectives:

1. Use more sustainable energy to power the Internet.
2. Reduce the Internet energy consumption for
   - running the network, *i.e.*, communicate; that is referred to as the operational cost;
   - producing the network, *i.e.*, build and renew hardware infrastructure; that is the embodied cost.

These two objectives are independent: harvesting and using sustainable energy is an overarching challenge; all fields and disciplines must improve in that dimension. However, **the best energy is the one we do not consume**. We must learn to use the energy that the Internet consumes better.

In this paper, we focus on reducing the Internet energy consumption. Previous works investigated several directions to achieve this goal, including designing an energy-aware API for cloud applications [7]; increasing the temperature in data centers to save on operational costs for cooling [21]; turning off devices to save energy in ISP [13] and datacenter networks [29]. This paper investigates two other research directions to save energy, which are often mentioned but have received surprisingly little concrete attention.

1. Reducing operational costs by using more energy-efficient routing protocols (§ 2).
2. Reducing embodied costs by keeping network devices alive longer (§ 3) which—counter-intuitively—might even improve the overall network reliability!

In this paper, we discuss why we believe these directions are promising, we sketch our initial thoughts to approach these problems, and we describe the set of practical challenges we foresee. In particular, we argue that we must rethink the codesign of routing protocols, networking hardware, and networking software to make the Internet more energy-efficient.

[1] Unformally defined as exceeding 5k servers and 1k square meters [2].

2 Cutting down operational costs

Almost two decades ago, Gupta and Singh [27] presented a simple calculation showing that transiting one byte over the Internet core consumes between 2× and 24× more energy than if using wireless LAN technology. While this was a rough estimation, and some numbers are probably outdated, this ballpark comparison most likely still holds; it might even be more favorable to wireless nowadays, as energy consumption generally increases with the Ethernet bandwidth.

We are not arguing that the Internet should become all wireless.[2] Instead, we seek to understand what causes the Internet to be so power-hungry and, most importantly, find ways to reduce its energy consumption.

2.1 Energy-inefficiencies of the Internet

Today, the energy-inefficiency of the Internet mainly stems from two facts.

Fact 1 Network devices are "always-on" but largely under-utilized because (i) ISPs design their network for peak traffic and (ii) aggressively over-provision to avoid congestion. Operators have reported upgrading links when utilization reaches as little as 50% [14, 39].

Fact 2 The energy consumed by today's networking devices is essentially independent of their load, as illustrated in Figure 1; *i.e.*, most of the cost comes from powering on the device (P_0), regardless of its utilization.

Based on these observations, a natural idea to improve energy efficiency is to keep devices off as much as possible and utilize the remaining ones more while powering on additional devices or links dynamically to adjust to the traffic demand. There has been some research in that direction, *e.g.*, [12, 13, 29, 38]; all these works concur that there is potential to save up to 50% of energy (see § 2.3). They also highlight three practical problems when turning devices on and off often.

Convergence Turning devices on and off creates instability and convergence issues in today's routing protocols and network systems (*e.g.*, optical signal amplifiers [42]).

Management Deciding which device to power off and when is a complex optimization problem which is hard to scale to run in real-time for large networks. Moreover, it requires accurate monitoring—or predictions—of the network traffic matrix, which is challenging in itself.

Start-up time Today's routers take minutes to boot [15, 29] which makes it unfeasible to power them on quickly to adjust to the traffic load. These long boot times are explained partly by the extensive memory testing commonly performed by networking devices [18].

[2] Although a recent work makes a good case for wireless ISPs [10].

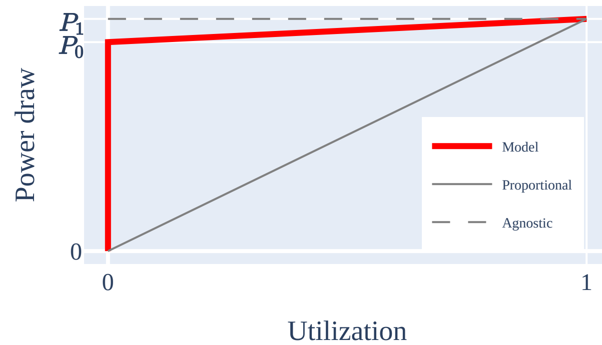

Figure 1: Typical power profile of network hardware. The largest portion of power consumption comes from powering on the hardware (P_0), which then increases only slightly with the utilization. For energy efficiency, the "ideal" power profile is proportional to the utilization. *Redrawn from [11].*

These are not significant problems as long as the "always-on" hypothesis holds. The Internet was designed under that assumption; energy efficiency was only a secondary objective, if at all. Some other networks were designed differently, though: *e.g.*, networked embedded systems' main requirement is energy efficiency, such as to provide long-term operations with only small batteries—or even without batteries [23].

2.2 Embedded-systems-inspired redesign

What if we redesigned Internet networks by taking inspiration from embedded systems?

Embedded systems is a field borne and grown with the mindset of managing energy scarcity. In 1999, the field was ushered with the vision of the "Smart Dust" [35], a large network of tiny devices embedding a small battery, some sensors, a solar panel to harvest energy, and a wireless transmitter and receiver; all that being highly energy-efficient to sustain autonomous operation. Two decades later, technology is getting ever closer to fulfilling that vision, thanks to:

Progress in hardware *e.g.*, non-volatile memory allows devices to maintain state over powered-off periods.

Progress in software *e.g.*, custom operating systems using less state, booting efficiently, and featuring multiple modes of operation to save power.

Progress in protocol design *e.g.*, reliable end-to-end communication over unreliable wireless links.

For wireless embedded systems, the radio is the largest energy consumer.[3] Thus, energy efficiency dictates turning the radio off for as much as possible—which is somewhat

[3] It was historically the case. Things are starting to change with transceivers getting more and more low-power and the push for more computing at the edge. But the point remains.

comparable to turning networking devices off (§ 2.1). Low-power wireless networking became a research field aiming to communicate efficiently between devices that are often "off" from the network's perspective. Many protocols have been designed, from fully centralized to distributed ones, with various cost-benefits trade-offs.

The protocol is (or should be) the starting point for an efficient system design, as it defines the resource requirements (*e.g.*, memory, compute) and the software abstractions needed to implement it. As system researchers, we are often constrained by the existing hardware and software stacks; we adapt our protocols or algorithms to be implementable on today's devices. These constraints fundamentally limit the achievable performance compared to a clean-slate redesign of protocol, hardware, and software.

To address the energy challenge in Internet networks, we argue that we must set these constraints aside for a moment. Instead of tweaking traditional routing protocols, as attempted in previous works [12, 27], we argue that improving the Internet energy efficiency is achievable with new protocols—designed under the assumption that devices will frequently turn themselves off—by taking inspiration from the vast literature on networked embedded systems. Naturally, it presents different design challenges and opportunities since the wireless and wireline physical layers are different. These new protocols will then lay out the practical challenges we must address on the hardware and software sides (see § 2.4).

2.3 How much is there to gain?

Let us estimate the potential gain of Internet routing redesign.

In a perfect world, network devices could be powered on/off instantaneously, and running the network at 100% utilization does not result in congestion or packet losses. Under these assumptions and considering the power model in Figure 1, we derive and compare the energy consumed by

- running a network at 100% utilization for short periods and turning all devices off the rest of the time;
- running the network at a baseline utilization U, always.

The potential savings depend on the baseline utilization U, as well as the P_0/P_1 ratio in the power model (see Figure 1). More precisely, the power draw $P(U)$ is given by

$$P(U) = P_0 + (P_1 - P_0) \cdot U \tag{1}$$

and the time taken to transmit B bytes on a link of capacity C with utilization U is

$$t(B,C,U) = B/(C \cdot U) \tag{2}$$

We can then compute the energy E_U and $E_{U_{max}}$ consumed to transmit at utilization U and $U_{max} = 1$, respectively, and the resulting energy savings $S = (E_U - E_{U_{max}})/E_U$.

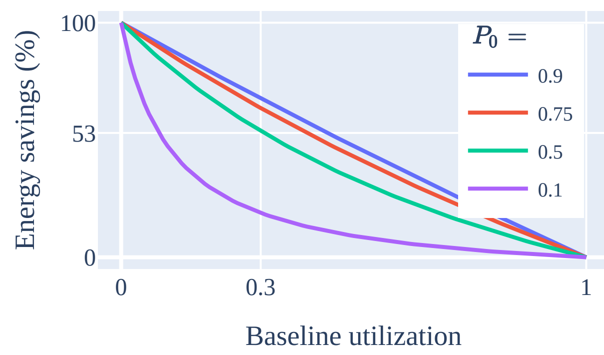

Figure 2: Even assuming that powering up a device accounts for only 50% of its maximum power draw ($P_0 = 0.5$), the potential energy savings are larger than 50% for baseline utilizations up to 30%. *Eq.* (5) *with* $P_1 = 1$.

$$\begin{aligned} E_U &= P(U) \cdot t(B,C,U) \\ &= B/C \cdot (P_0/U + P_1 - P_0) \end{aligned} \tag{3}$$

$$\begin{aligned} E_{U_{max}} &= P_1 \cdot t(B,C,1) \\ &= P_1 \cdot B/C \end{aligned} \tag{4}$$

$$\Rightarrow \qquad S = \frac{P_0 \cdot (1-U)}{P_0 + (P_1 - P_0) \cdot U} \tag{5}$$

Figure 2 shows the savings from Eq. (5) with $P_1 = 1$ and different values for P_0. As expected, the smaller the baseline utilization, the more potential savings; if traffic demands an average utilization of 99%, there is little margin to turn devices off. However, given the state of practice in network capacity planning [14, 39], we rather expect the average utilization in ISP networks today to be in the low tens of percent. Moreover, the larger P_0/P_1, the more potential savings; according to the literature, we expect this ratio to be at most 0.5 for standard devices [11, 13, 29]. Assuming a baseline utilization below 30%, this yields $\geq 50\%$ energy savings!

Naturally, this is a rough approximation of the potential savings, but it provides a useful upper bound of what one can hope to achieve. It also highlights some practical research challenges to realize such savings, *e.g.*, fast powering-on/off times and running the network reliably at high utilization.

2.4 Practical challenges

Given the power profile of today's network devices (Figure 1), it is clear that turning them off whenever possible[4] has some potential to reduce the Internet operational costs significantly (§ 2.3). However, achieving this potential requires reducing the start-up time of networking devices.

[4]Or turning off only certain components [13]

More precisely, to harness the energy benefits without dramatically increasing delay, we must bring the time for a device to go from a low-power mode to "ready to forward traffic" down from the *minute* scale to the *millisecond* scale. We believe this is possible, but it will demand redesigning the networking hardware, software, and protocols. Here are some of the directions we are exploring:

Redesign routing protocols

- to be more distributed and cope better with network nodes switching on/off often;
- to require less state in network devices, hence requiring less memory;[5]
- to tolerate transient faults resulting, *e.g.*, of using an approximate forwarding state.

Redesign networking hardware

- to include non-volatile memory and speed up the state reconstruction after powering on;
- to improve power proportionality by power-gating peripherals, using DVFS[6], or running RAPL.[7] [8]

Redesign networking software

- to extend wake-on-LAN [6] to networking devices;
- to optimize the boot time of network devices' OS.

It is an ambitious plan, but the experience of embedded systems has shown that co-designing hardware, software, and protocols can push the energy efficiency of the overall system design very far. We strongly believe that the wireline networking community must follow that example and redesign the Internet to make it more energy-efficient.

3 Cutting down embodied costs

The more we cut down the operational costs, the larger becomes the remaining share: *i.e.*, the embodied costs, the carbon impact due to the manufacturing and disposal of devices. Recent studies argue that 52% of carbon emissions from laptops are embodied [41]—that is, it costs *more* to produce them than to use them. Today, this share is estimated between 10 and 20% for servers [1, 41]. Concretely, this means we cannot just count on next-generation hardware to save energy since the production of these new devices leads to *more* emissions that will become increasingly harder to offset with operational gains.

The solution to reducing the embodied costs is simple: keep network devices alive longer. Nowadays, many organizations refresh their networking infrastructure every 3–5 years [32]. These numbers are similar to the refresh cycles of ICT equipment in the datacenter industry [4].

[5] *E.g.*, using source-based routing mechanisms as explored in [33]

[6] Dynamic Voltage and Frequency Scaling

[7] Running Average Power Limit

[8] These technologies are now common on servers. Could they be transferred to networking devices, and how much is there to gain with those?

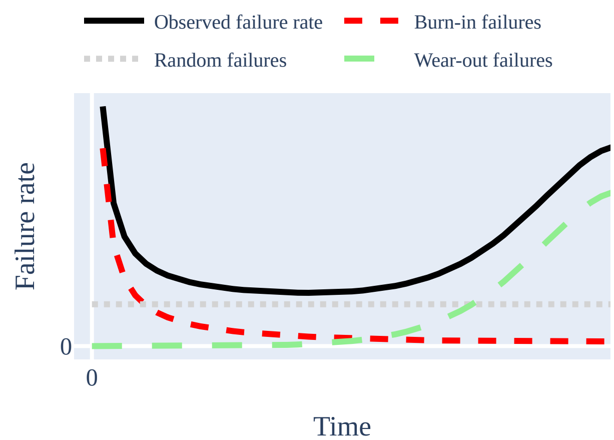

Figure 3: Devices failures rates result from three effects: manufacturing defects (burn-in), random failures, and wear-out (*i.e.*, aging). *In reliability engineering, the modeling and analysis of failure probabilities are often based on the Weibull distribution* [5].

A priori, keeping network devices alive longer may seem like a bad deal as it would lead to networks that are:

Less reliable due to increased failures rates;
Less secure due to the eventual lack of vendor support;
Harder to manage due to increased device diversity.

We argue against this intuition: "older" networks do not have to compromise on reliability, security, or ease of management—as long as they are designed and managed with longevity in mind. We even believe that "older" networks could perform even better than freshly-deployed ones.

"Older" networks are *not* necessarily less reliable Perhaps counter-intuitively, it is well-known in reliability engineering that new devices often suffer from manufacturing defects early in their lifetime. As observed by [17]: *"The vast majority of network hardware failures take place within the first 30 days of installing brand new, out-of-the-box network hardware."* After this "burn-in" period, the failure rate decreases to reach a plateau (corresponding to random failures) before rising again due to aging and wearing. When compounded over the device lifetime, these three effects lead to so-called "bathtub" curves [3] (see Figure 3).

"Bathtub" curves explain why renewing devices can decrease the reliability of a network, at least provided that the older devices are not yet worn-out. We believe that the delay required to observe these high worn-out-induced failure rates might be considerably longer than 3–5 years. We see two hints of that: First, network vendors including Cisco [16], Juniper Networks [34], and HP [30] ensure a *5-year* window between the end-of-sale and the end-of-support. Since devices are typically sold for several years before reaching end-of-sale, network devices often have close to a decade of vendor support. Second, some companies specialize in refurbishing network hardware [17]—some even having a lifetime warranty[9]—suggesting that "old" network equipment is perfectly functional way-passed its proclaimed obsolescence.

Reasoning about how much "older" can a network device get before it is time for a change requires instantiating the model shown in Figure 3 according to device characteristics and network operation modes (*e.g.*, the average temperature the device operates in [21]). To the best of our knowledge, such models are not (yet) well-established in networking. In particular, we lack measurements of the failure rates over long periods (years)— [26] is one of few. We believe this is a worthy goal for the research community to tackle. With such a model, one can use seasoned methods from reliability engineering to compute the expected MTTF (mean time to failure) or MTBF (mean time between failures) of devices and devise a rational plan for hardware maintenance and replacement.

An "older" network is *not* necessarily less secure The relationship between how old a device is and how secure it is is unclear and calls for more research. On the one hand, one could argue that running tried-and-tested equipment instead of bleeding-edge one can decrease the number of vulnerabilities, as new devices often ship with new hardware and software codebases. On the other hand, maintaining devices passed their official end-of-life dates means that no one will patch newly-found vulnerabilities for the old codebases. We envision two pragmatic solutions to address this:

1. Incentivize network vendors to extend end-of-life policy;
2. Incentivize network operators to rely on "white-box" equipment and open-source software stacks allowing anyone to provide support for a given device.

An "older" network is *not* necessarily harder to manage We believe that the recent successes in declarative network management—such as configuration verification [8, 22, 24, 36, 40] and synthesis [9, 19, 20]—can quickly mitigate the complexity induced by a higher device heterogeneity. In fact, recent work from Google [25] fully embraces the idea of mixing different network technology within one datacenter.

Ultimately, the networking community should address the following questions: When does it make sense to renew networking hardware? What are the practical consequences of operating older devices? When do aging effects appear? Or course, hardware might also be renewed because more "powerful" devices are required. However, given the weight of the embodied costs in networking, it is crucial that the refresh cycle policies of tomorrow's networks[10] are driven by rational science and purpose—not marketing.

[9] www.cxtec.com/hardware/equal2new/lifetime-warranty/

[10] Taking into account the operational costs of newer versus older devices.

4 Conclusions

Given the global sustainability challenge that humanity is facing, the energy inefficiency of the Internet appears—more and more—as unacceptable. In this paper, we discuss two research directions that have the potential to reduce the carbon footprint of Internet networks significantly, namely (i) redesigning routing protocols to save energy by putting under-utilized devices to sleep (§ 2); (ii) reducing the embodied carbon of hardware by keeping them running longer (§ 3).

Realizing these ideas requires a lot of research, including redesigning networking hardware and software to be more energy-efficient; understanding better "where power goes?" in today's devices; studying the aging of networking hardware in different utilization conditions (*e.g.*, How does turning devices on/off often affects aging?). This is an enormous effort that the entire networking community must tackle.

References

[1] HPE product carbon footprint – HPE ProLiant DL360 Gen10 server data sheet. URL: https://www.hpe.com/psnow/doc/a50002430enw.

[2] What Is a Hyperscale Data Center? URL: https://www.vertiv.com/en-asia/about/news-and-insights/articles/educational-articles/what-is-a-hyperscale-data-center/.

[3] Bathtub curve. *Wikipedia*, May 2021. URL: https://en.wikipedia.org/w/index.php?title=Bathtub_curve&oldid=1024297779.

[4] Analyzing hardware refresh cycles in the data center, 2022. URL: https://horizontechnology.com/news/data-center-hardware-refresh-cycles/.

[5] Weibull distribution. *Wikipedia*, April 2022. URL: https://en.wikipedia.org/w/index.php?title=Weibull_distribution&oldid=1085195154.

[6] AMD. Magic Packet Technology, November 2015. URL: https://www.amd.com/system/files/TechDocs/20213.pdf.

[7] Thomas Anderson, Adam Belay, Mosharaf Chowdhury, Asaf Cidon, and Irene Zhang. Treehouse: A Case For Carbon-Aware Datacenter Software. *arXiv:2201.02120 [cs]*, January 2022. URL: http://arxiv.org/abs/2201.02120, arXiv:2201.02120.

[8] Ryan Beckett, Aarti Gupta, Ratul Mahajan, and David Walker. A General Approach to Network Configuration Verification. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pages 155–168, Los Angeles CA USA, August 2017.

ACM. URL: https://dl.acm.org/doi/10.1145/3098822.3098834, doi:10.1145/3098822.3098834.

[9] Ryan Beckett, Ratul Mahajan, Todd Millstein, Jitendra Padhye, and David Walker. Don't Mind the Gap: Bridging Network-wide Objectives and Device-level Configurations. In *Proceedings of the 2016 ACM SIGCOMM Conference*, SIGCOMM '16, pages 328–341, New York, NY, USA, August 2016. Association for Computing Machinery. doi:10.1145/2934872.2934909.

[10] Debopam Bhattacherjee, Waqar Aqeel, Sangeetha Abdu Jyothi, Ilker Nadi Bozkurt, William Sentosa, Muhammad Tirmazi, Anthony Aguirre, Balakrishnan Chandrasekaran, P. Brighten Godfrey, Gregory Laughlin, Bruce Maggs, and Ankit Singla. cISP: A Speed-of-Light Internet Service Provider. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, pages 1115–1133, 2022. URL: https://www.usenix.org/conference/nsdi22/presentation/bhattacherjee.

[11] Aruna Prem Bianzino, Claude Chaudet, Federico Larroca, Dario Rossi, and Jean-Louis Rougier. Energy-aware routing: A reality check. In *2010 IEEE Globecom Workshops*, pages 1422–1427, December 2010. doi:10.1109/GLOCOMW.2010.5700172.

[12] Aruna Prem Bianzino, Luca Chiaraviglio, Marco Mellia, and Jean-Louis Rougier. GRiDA: GReen Distributed Algorithm for energy-efficient IP backbone networks. *Computer Networks*, 56(14):3219–3232, September 2012. URL: https://www.sciencedirect.com/science/article/pii/S1389128612002344, doi:10.1016/j.comnet.2012.06.011.

[13] J. Chabarek, J. Sommers, P. Barford, C. Estan, D. Tsiang, and S. Wright. Power Awareness in Network Design and Routing. In *IEEE INFOCOM 2008 - The 27th Conference on Computer Communications*, pages 457–465, April 2008. doi:10.1109/INFOCOM.2008.93.

[14] Cisco. Cisco WAN Automation Engine (WAE) - Best Practices in Core Network Capacity Planning White Paper. URL: https://www.cisco.com/c/en/us/products/collateral/routers/wan-automation-engine/white_paper_c11-728551.html.

[15] Cisco. Cisco 3750, 2900, 3900 and 4400 Boot Time, November 2018. URL: https://community.cisco.com/t5/switching/cisco-3750-2900-3900-and-4400-boot-time/td-p/3739176.

[16] Cisco. End-of-life policy, 2022. URL: https://www.cisco.com/c/en/us/products/contact-center/eos-eol-policy.html.

[17] CXTEC. Surprising truth about network hardware Failures, March 2022. URL: https://www.cxtec.com/blog/network-hardware-failures-shocking-truth/.

[18] Dell. POST and Boot Processes. URL: https://www.dell.com/support/kbdoc/en-uk/000128270/post-and-boot-procedures?lwp=rt.

[19] Ahmed El-Hassany, Petar Tsankov, Laurent Vanbever, and Martin Vechev. Network-Wide Configuration Synthesis. In Rupak Majumdar and Viktor Kunčak, editors, *Computer Aided Verification*, Lecture Notes in Computer Science, pages 261–281, Cham, 2017. Springer, Springer International Publishing. doi:10.1007/978-3-319-63390-9_14.

[20] Ahmed El-Hassany, Petar Tsankov, Laurent Vanbever, and Martin Vechev. Netcomplete: Practical network-wide configuration synthesis with autocompletion. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*, pages 579–594, 2018. URL: https://www.usenix.org/conference/nsdi18/presentation/el-hassany.

[21] Nosayba El-Sayed, Ioan A. Stefanovici, George Amvrosiadis, Andy A. Hwang, and Bianca Schroeder. Temperature management in data centers: Why some (might) like it hot. In *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '12, pages 163–174, New York, NY, USA, June 2012. Association for Computing Machinery. doi:10.1145/2254756.2254778.

[22] Ari Fogel, Stanley Fung, Luis Pedrosa, Meg Walraed-Sullivan, Ramesh Govindan, Ratul Mahajan, and Todd Millstein. A General Approach to Network Configuration Analysis. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*, pages 469–483, 2015. URL: https://www.usenix.org/conference/nsdi15/technical-sessions/presentation/fogel.

[23] Kai Geissdoerfer and Marco Zimmerling. Learning to Communicate Effectively Between Battery-free Devices. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, pages 419–435, 2022. URL: https://www.usenix.org/conference/nsdi22/presentation/geissdoerfer.

[24] Nick Giannarakis, Devon Loehr, Ryan Beckett, and David Walker. NV: An intermediate language for verification of network control planes. In *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI

2020, pages 958–973, New York, NY, USA, June 2020. Association for Computing Machinery. doi:10.1145/3385412.3386019.

[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. Aquila: A unified, low-latency fabric for datacenter networks. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, pages 1249–1266, 2022. URL: https://www.usenix.org/conference/nsdi22/presentation/gibson.

[26] Phillipa Gill, Navendu Jain, and Nachiappan Nagappan. Understanding network failures in data centers: Measurement, analysis, and implications. In *Proceedings of the ACM SIGCOMM 2011 Conference*, SIGCOMM '11, pages 350–361, New York, NY, USA, August 2011. Association for Computing Machinery. doi:10.1145/2018436.2018477.

[27] Maruti Gupta and Suresh Singh. Greening of the internet. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, pages 19–26, New York, NY, USA, August 2003. Association for Computing Machinery. doi:10.1145/863955.863959.

[28] M. Handley. Why the Internet only just works. *BT Technology Journal*, 24(3):119–129, July 2006. doi:10.1007/s10550-006-0084-z.

[29] Brandon Heller, Srini Seetharaman, Priya Mahadevan, Yiannis Yiakoumis, Puneet Sharma, Sujata Banerjee, and Nick McKeown. ElasticTree: Saving energy in data center networks. In *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation*, NSDI'10, page 17, USA, April 2010. USENIX Association. URL: https://www.usenix.org/legacy/event/nsdi10/tech/full_papers/heller.pdf.

[30] HP. End-of-life products, 2022. URL: https://techlibrary.hpe.com/de/de/networking/products/eos/information.aspx.

[31] Hugues Ferreboeuf. Lean ICT: Towards digital sobriety. Technical report, Paris, March 2019. URL: https://theshiftproject.org/wp-content/uploads/2019/03/Executive-Summary_Lean-ICT-Report_EN_lowdef.pdf.

[32] icorps Technologies. Useful life of IT network equipment: Assets & perspective, 2015. URL: https://blog.icorps.com/determining-the-useful-life-of-your-it-network.

[33] Seng-Kyoun Jo, Lin Wang, Jussi Kangasharju, and Max Mülhäuser. Eco-friendly Caching and Forwarding in Named Data Networking. In *2020 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN*, pages 1–6, July 2020. doi:10.1109/LANMAN49260.2020.9153230.

[34] Juniper Networks. Product end-of-life policy, 2020. URL: https://support.juniper.net/support/pdf/eol/990833.pdf.

[35] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Next century challenges: Mobile networking for"Smart Dust". In *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, MobiCom '99, pages 271–278, New York, NY, USA, August 1999. Association for Computing Machinery. doi:10.1145/313451.313558.

[36] Peyman Kazemian, George Varghese, and Nick McKeown. Header Space Analysis: Static Checking for Networks. In *9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*, pages 113–126, 2012. URL: https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/kazemian.

[37] Ray Le Maistre. How many hyperscale data centres does the world need? Hundreds more, it seems, March 2022. URL: https://www.telecomtv.com/content/digital-platforms-services/how-many-hyperscale-data-centres-does-the-world-need-hundreds-more-it-seems-44015/.

[38] Sergiu Nedevschi, Lucian Popa, Gianluca Iannaccone, Sylvia Ratnasamy, and David Wetherall. Reducing Network Energy Consumption via Sleeping and Rate-Adaptation. In *5th USENIX Symposium on Networked Systems Design and Implementation (NSDI 08)*, 2008. URL: https://www.usenix.org/conference/nsdi-08/reducing-network-energy-consumption-sleeping-and-rate-adaptation.

[39] Hank Nussbacher. Nanog: Bottlenecks and link upgrades. URL: https://seclists.org/nanog/2020/Aug/193.

[40] Konstantin Weitz, Doug Woos, Emina Torlak, Michael D. Ernst, Arvind Krishnamurthy, and Zachary Tatlock. Scalable verification of border gateway protocol configurations with an SMT solver. In *Proceedings of the 2016 ACM SIGPLAN International Conference*

on Object-Oriented Programming, Systems, Languages, and Applications, OOPSLA 2016, pages 765–780, New York, NY, USA, October 2016. Association for Computing Machinery. doi:10.1145/2983990.2984012.

[41] Wim Vanderbauwhede. Frugal computing, 2021. URL: https://wimvanderbauwhede.github.io/articles/frugal-computing/.

[42] Zhizhen Zhong, Manya Ghobadi, Maximilian Balandat, Sanjeevkumar Katti, Abbas Kazerouni, Jonathan Leach, Mark McKillop, and Ying Zhang. BOW: First Real-World Demonstration of a Bayesian Optimization System for Wavelength Reconfiguration. In *Optical Fiber Communication Conference (OFC) 2021*, page F3B.1, Washington, DC, 2021. OSA. URL: https://opg.optica.org/abstract.cfm?URI=OFC-2021-F3B.1, doi:10.1364/OFC.2021.F3B.1.