OpenInfra: A Co-simulation Framework for the Infrastructure Nexus

Jiaheng Lu, Yunming Xiao, Shmeelok Chakraborty, Silvery Fu[‡], Yoon Sung Ji,
Ang Chen, Mosharaf Chowdhury, Nalini Rao[§], Sylvia Ratnasamy[‡], Xinyu Wang
University of Michigan [‡]UC Berkeley [§]EPRI

1 Introduction

Industrial infrastructures (e.g., datacenters, power grids, and water systems) are interdependent upon each other [27]. Datacenters need electricity from power grids and large amounts of water for cooling. Similarly, power and water systems have tight interactions—e.g., water pumps require electricity, and power plants need water for cooling. Further, each type of infrastructure is itself a federation of interdependent subsystems; for instance, power plants may collaboratively generate electricity and their operations are interdependent. Civil engineering researchers aptly use the term "nexus" to describe the intricate relation within and across industrial infrastructures [35], and the infrastructure nexus is an increasingly important interdisciplinary field [40].

This study, however, has not yet attracted sufficient attention from the computing community—despite that datacenters are a major infrastructure with an outsized impact, and that the "datacenter/X" nexus (e.g., datacenter/energy nexus [15, 25], datacenter/water nexus [26, 52]) have repeatedly made the headlines [5, 7, 11]. In 2018, data centers consumed 205 TWh of electricity, more than the annual usage of countries like Ireland and Denmark [36, 43]. By 2030, they are projected to use 10% of the world's electricity [28, 32, 37, 41]. Moreover, the datacenter itself is a nexus of computing, electricity, and cooling subsystems—e.g., 48% of their energy is used by non-IT equipment, with 79% of that going to cooling [39].

Given increasingly tight interactions between infrastructures, co-optimization becomes crucial and involves navigating different tradeoffs in the design space. For example, cooling datacenters at night is more water-efficient [1, 31] but increases carbon emissions due to lower renewable energy availability [13]. Since direct experimentation on multiple physical infrastructures is difficult, it is important for the computing community to develop simulation support that enables testing across these systems, i.e., at the infrastructure nexus.

We have been investigating *software simulation* support for the infrastructure nexus, including but going beyond datacenters. Simulation is the de-facto approach to infrastructure-scale studies. As a case in point, datacenter researchers routinely use networking, server-level, and datacenter-level simulators, such as OMNet++ [10], CloudSim [6], and EnergyPlus [19], for experimentation. Similarly, researchers rely on simulators for other types of industrial infrastructures at scale—e.g., Open5GS [9] and free5GC [8] for 5G deployments, Carbon Explorer [13], Ecovisor [45] and Vessim [50] for renewable energy resources. However, despite the extensive use of simulation, designing and implementing simulators for the *infrastructure nexus* raises unique challenges and remains underexplored.

Building high-fidelity simulators for industrial infrastructures requires deep domain expertise, which is fragmented across different infrastructure sectors (e.g., power vs. water vs. datacenters). Hence, no single community of researchers is well placed to develop a simulator that adequately addresses the challenges of multiple large infrastructures. Moreover, across infrastructure communities, existing simulators work in very different ways. Some focus on device-level simulation (e.g., switches, servers) [10] whereas others simulate at the infrastructure level (e.g., entire power plants) [13]. From the software perspective, they also provide different interfaces for simulation and operate at varying timescales. End-to-end simulation across infrastructures, therefore, is a systems design problem, and it requires tackling software-level challenges.

We are developing OpenInfra, a co-simulation software system for researchers that study the infrastructure nexus, including the "datacenter/X" nexus and more—e.g., compute/power/water devices within datacenters, or across power/water systems. Co-simulation is a technique where global simulation of a coupled system can be achieved by composing individual simulation of its parts [17, 18, 20, 24, 33]. OpenInfra aims to provide end-to-end simulation with a holistic view, enabling studies that individual simulators cannot easily achieve. For instance, consider two example nexus studies:

- **Datacenter/X nexus:** To date, numerous simulation frameworks excel at modeling individual systems like datacenters [4], water cooling systems [42], and power grids [21]. Recent efforts have expanded to include datacenter simulations with integrated power supply and chiller systems. However, a significant gap remains in co-simulating these interdependent subsystems. The failure of one system, such as the power grid, can trigger a domino effect impacting the water cooling system and datacenter—issues that individual simulators cannot fully capture.
- **Power/water nexus:** The interaction between power grids and water systems is a critical area of study. Power generation equipment, such as turbines and generators, requires cooling (often using water) to prevent overheating and maintain efficiency [49]. Water cooling systems also

The 2nd Workshop on Hot Topics in System Infrastructure (HotInfra'24), co-located with SOSP'24, November 3, 2024, Austin, TX, USA

help manage heat discharge in large infrastructures. Additionally, pumps in water distribution networks consume energy while supplying water for power plant cooling. In hydroelectric power, water is stored in dams and later released to generate electricity.

Our goal for designing this framework is to take a modular approach, where individual simulators can be *plugged* into our framework (mostly) as-is, allowing the framework to focus on simulating their nexus/interactions. This not only enables reuse of existing domain-specific simulators developed by domain experts, but also allows for the integration of new ones over time. While this is still work-in-progress, we describe the system-level challenges in stitching together disparate simulators that we have identified thus far:

- *Language:* Individual simulators are implemented in different languages, but OpenInfra itself should expose an intuitive language for programmers to define a nexus experiment—e.g., studying how water distribution networks consuming energy. This language is ideally declarative and easy to use.
- *Execution:* Co-simulation performance is constrained by the slowest underlying simulator, particularly discrete event simulations [51], and memory- or compute-intensive Machine Learning (ML) model inference-based simulators like m3 [12]. Identifying opportunities for parallel execution across simulators will be essential.
- *Scalability:* As the number of simulators grows, hosting them on a single machine may become challenging. On the other hand, exchanging simulator state across servers leads to communication overhead. OpenInfra should identify optimal placements of simulators and efficient state transfer methods.
- *Algorithm:* The co-simulation program must be sufficiently expressive to enable many use cases, and it may need to integrate with existing libraries (e.g., ML libraries for predictive analysis, or linear/non-linear optimization libraries) for various simulation goals.
- *Synchronization:* Most events within a simulator require no external visibility, but certain events will trigger processing in OpenInfra co-simulation program and in turn, additional events in other simulators. Hence, OpenInfra needs to synchronize and translate events across simulation boundaries in a semantically meaningful manner.
- *Data management:* Each simulator has built-in data formats, e.g., for defining events, traces, and observations, whereas the nexus simulation requires understanding across these internal data traces. OpenInfra therefore needs to handle data management challenges, such as integrating diverse data formats and designing conversion adapters.

OpenInfra is written in Python with 4,000 lines of code. It is open-source and available on GitHub [1] .

[1] https://github.com/JhengLu/OpenInfra

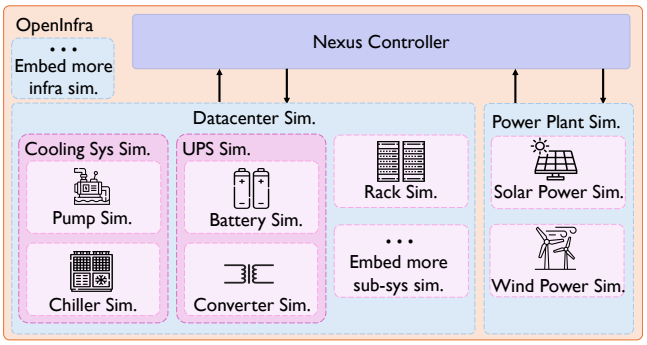

Figure 1. Structure of OpenInfra.

2 Initial Exploration

Our current design and implementation are informed by the challenges outlined above, and we now describe the design choices that we have encountered. As Figure 1 shows, OpenInfra is a framework for embedding diverse simulators, and we call the co-simulation program the "nexus controller."

Simulator Wrapper: We found that, although our integration has not required extensive modification to individual simulators, it has been necessary to rearchitect clean interfaces to and from each underlying simulator. Thus far, we have been adopting the approach of reengineering wrappers around simulators to expose two types of APIs: *State()* APIs expose state variables internal to a simulator (e.g., the amount of generated energy from solar/wind farms) to OpenInfra, and *Actuate()* APIs trigger certain processing inside a simulator (e.g., turning on backup battery due to power shortage).

Case Study: Consider how OpenInfra has integrated an uninterruptible power supply (UPS) simulator. A UPS device is commonly used in datacenters to supply power to individual racks. A UPS contains a battery with backup energy. The state of charge (SoC) of the battery is managed in its simulator (e.g., PyBaMM), and we have engineered an *Actuate()* API that updates the battery's charge state by passing initial SoC, power, and duration to the simulator, along with a *State()* API to query the updated SoC. In turn, OpenInfra feeds this information to other simulators that require information about the SoC. These APIs hide complexity inside the simulators, and allow for potentially integrating "hardware-in-the-loop" (HITL) tests. For instance, a rack simulator that exposes the amount of consumed energy to OpenInfra could be replaced by actual calls into individual hardware servers to obtain real-time data. In a similar vein, simulators could perform trace-driven simulation relying on publicly available traces (e.g., energy traces released by Google [44]). The traces will be preprocessed to ensure consistent simulator output, standardizing aspects like time steps and units.

Design Principle: In designing OpenInfra, we follow the principles of Infrastructure-as-Code (IaC) [14], a popular paradigm for configuring cloud resources. Using IaC terms, each simulator is defined by a configuration file as a "resource," and the overall simulation is a hierarchy of resources

forming a dependency graph. Users therefore can flexibly define the topology of infrastructures and interactions, and the nexus controller will instantiate the individual simulators based on the IaC configuration automatically. To synchronize multiple simulators, OpenInfra currently advances the time using the smallest time step among all the simulators. The controller stores events, matches them by type, and translates them into corresponding events for other simulators. It can also create multiple simulator-level events (e.g., failures). Currently, the nexus controller follows a fixed schedule to invoke simulators at each time step, but an interesting topic of study is investigating how to better parallelize the execution of individual simulators for higher performance. Integrating results from different simulators can be challenging due to variations in data formats. We plan to develop a Retrieval-Augmented Generation (RAG) [30] agent (e.g., using LlamaIndex[34]) that can read the entire documentation of a simulator while digesting its traces.

Supported Simulators: Our prototype integrates 15 high-fidelity simulators. OpenInfra includes a Google trace-driven simulator [48] to model CPU utilization across servers, with power consumption simulated using SPEC Power Benchmark data [46] and interpolated based on CPU utilization [22, 44]. The system also features a power plant simulator using EIA data [2], supporting eight types of power plants. For UPS batteries, OpenInfra deploys two lithium-ion battery simulators [13, 29] and interfaces with the advanced PyBaMM simulator [47]. Additionally, OpenInfra includes a pump simulator based on [23] and a chiller model from H2P [52]. Water usage in the datacenter is simulated using the water usage effectiveness (WUE) metric [1, 3, 27, 38], calculated by dividing the liters of water used for humidification and cooling by the total annual power required to operate the IT equipment. Water usage for cooling the power plant is calculated using the Energy Water Intensity Factor (EWIF) [16], which indicates the amount of water required to produce a specific amount of energy. Moving forward, we plan to incorporate 5G simulators and refine our interfaces to support more advanced simulators for existing components.

3 Evaluation

We simulated a data center for 100 hours with 5 UPS units and 7,392 servers. The WUE value was set at 1.6 L/kWh, with EWIF for non-renewable power at 0.8 L/kWh and zero for renewable power. Total power usage includes consumption by all IT and non-IT equipment. Data center water usage was calculated using a WUE-based simulator, while power plant water usage was determined through an EWIF-based simulator. Server load follows the Google trace [48], and power plant data is sourced from the EIA [2]. Both traces start at midnight. We also implemented a carbon-aware scheduling algorithm that adjusts the server load based on the availability of renewable power and battery storage.

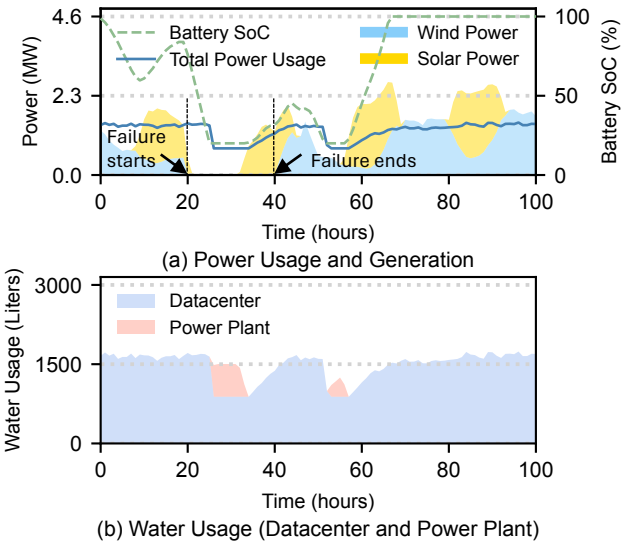

Figure 2. Simulation of infrastructure nexus: wind power plant failure from 20h to 40h.

Figure 2 illustrates our simulation. Figure 2(a) presents the power consumption, the renewable power supply (with wind and solar energy displayed as stacked components), and the battery state of charge (SoC) serving as a buffer between consumption and supply. Figure 2(b) illustrates the water usage of the datacenter and the power plant. The datacenter's water usage, which is for cooling the servers, is directly proportional to the server workloads and, consequently, to the total power usage. In contrast, the power plant's water usage is associated with non-renewable energy generation and is directly proportional to the amount of power produced by non-renewable sources (not depicted in Figure 2(a)).

Between 0-10h, power consumption exceeds the renewable supply, leading to a drop in the battery SoC as it compensates for the shortfall. From 10-20h, the renewable supply surpasses consumption, allowing the battery to recharge. However, at 20h, we simulate a wind power generator failure that lasts until 40h. During 20-25h, solar power is also unavailable due to nighttime, so the battery alone maintains the power supply. At 25h, the battery SoC drops below a preset threshold, necessitating a reduction in server workload and the activation of non-renewable energy sources. Consequently, power usage in Figure 2(a) is nearly halved, reducing the datacenter's water usage in Figure 2(b). Meanwhile, water usage at the power plant increases due to the reliance on non-renewable energy. At 32h, solar power recovers, allowing the power plant to cease non-renewable energy generation, and the datacenter workload begins to recover. The wind power failure ends at 40h, resulting in total power supply exceeding consumption once again. A similar power supply drop occurs around 50h, but after 55h, the supply stabilizes and the battery recharges to full capacity.

Acknowledgements

We thank the reviewers for their feedback. This work was partially supported by the Bold Challenges Accelerate Program at the University of Michigan.

References

[1] How microsoft measures datacenter water and energy use to improve azure cloud sustainability. https://azure.microsoft.com/en-us/blog/how-microsoft-measures-datacenter-water-and-energy-use-to-improve-azure-cloud-sustainability/.

[2] How much electricity does an american home use? https://www.eia.gov/tools/faqs/faq.php?id=97&t=3.

[3] Meta 2023 sustainability report. https://sustainability.atmeta.com/wp-content/uploads/2023/07/Meta-2023-Sustainability-Report-1.pdf.

[4] Simgrid: A toolkit for the simulation of application scheduling. In *Proceedings First IEEE/ACM International Symposium on Cluster Computing and the Grid*, pages 430–437. IEEE, 2001.

[5] As the world goes digital, datacenters that make the cloud work look to renewable energy sources. 2024. https://news.microsoft.com/source/emea/features/as-the-world-goes-digital-datacenters-that-make-the-cloud-work-look-to-renewable-energy-sources/.

[6] CloudSim: A Framework for Modeling and Simulation of Cloud Computing Infrastructures and Services. 2024. http://www.cloudbus.org/cloudsim/.

[7] Data center water usage: A comprehensive guide. 2024. https://dgtlinfra.com/data-center-water-usage/.

[8] free5GC: an open-sourec 5G core network. 2024. https://www.free5gc.org/.

[9] Open5gs. 2024. https://open5gs.org/.

[10] Opensim. omnet++. 2024. https://www.omnetpp.org.

[11] US tech groups' water consumption soars in 'data centre alley'. 2024. https://www.ft.com/content/1d468bd2-6712-4cdd-ac71-21e0ace2d048?segmentId=2c1df321-36a4-1206-2c08-112c059dd69d.

[12] C. L. , A. Nasr-Esfahany, K. Z. , K. Noorbakhsh, P. Goyal, M. Alizadeh, and T. Anderson. m3: Accurate flow-level performance estimation using machine learning. In *ACM SIGCOMM*, August 2024.

[13] B. Acun, B. Lee, F. Kazhamiaka, K. Maeng, U. Gupta, M. Chakkaravarthy, D. Brooks, and C.-J. Wu. Carbon explorer: A holistic framework for designing carbon aware datacenters. In *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, pages 118–132, 2023.

[14] M. Artac, T. Borovssak, E. Di Nitto, M. Guerriero, and D. A. Tamburri. Devops: introducing infrastructure-as-code. In *Proceedings of the 2017 IEEE/ACM International Conference on Software Engineering Companion*, 2017.

[15] J. Athavale, C. Bash, W. Brewer, M. Maiterth, D. Milojicic, H. Petty, and S. Sarkar. Digital twins for data centers. *Computer*, 57(10):151–158, 2024.

[16] D. Azevedo, S. C. Belady, and J. Pouchet. Water usage effectiveness (wue): A green grid datacenter sustainability metric. *The Green Grid*, 32, 2011.

[17] D. Broman, C. Brooks, L. Greenberg, E. A. Lee, M. Masin, S. Tripakis, and M. Wetter. Determinate composition of fmus for co-simulation. In *2013 Proceedings of the International Conference on Embedded Software (EMSOFT)*, pages 1–12. IEEE, 2013.

[18] S. Cho, M. Patel, H. Chen, M. Ferdman, and P. Milder. A full-system vm-hdl co-simulation framework for servers with pcie-connected fpgas. In *Proceedings of the 2018 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pages 87–96, 2018.

[19] D. B. Crawley, L. K. Lawrie, F. C. Winkelmann, W. F. Buhl, Y. J. Huang, C. O. Pedersen, R. K. Strand, R. J. Liesen, D. E. Fisher, M. J. Witte, et al. Energyplus: creating a new-generation building energy simulation program. *Energy and buildings*, 33(4):319–331, 2001.

[20] F. Cremona, M. Lohstroh, D. Broman, E. A. Lee, M. Masin, and S. Tripakis. Hybrid co-simulation: it's about time. *Software & Systems Modeling*, 18:1655–1679, 2019.

[21] W. Dong, Z. Xie, G. Kestor, and D. Li. Smart-pgsim: Using neural network to accelerate ac-opf power grid simulation. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15. IEEE, 2020.

[22] X. Fan, W.-D. Weber, and L. A. Barroso. Power provisioning for a warehouse-sized computer. *ACM SIGARCH computer architecture news*, 35(2):13–23, 2007.

[23] A. A. Ghoneim. Design optimization of photovoltaic powered water pumping systems. *Energy conversion and management*, 47(11-12):1449–1463, 2006.

[24] C. Gomes, C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe. Co-simulation: a survey. *ACM Computing Surveys (CSUR)*, 51(3):1–33, 2018.

[25] U. Gupta, Y. G. Kim, S. Lee, J. Tse, H.-H. S. Lee, G.-Y. Wei, D. Brooks, and C.-J. Wu. Chasing carbon: The elusive environmental footprint of computing. In *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 854–867. IEEE, 2021.

[26] M. Hogan. Data flows and water woes: The utah data center. *Big Data & Society*, 2(2):2053951715592429, 2015.

[27] M. Jalili, I. Manousakis, Í. Goiri, P. A. Misra, A. Raniwala, H. Alissa, B. Ramakrishnan, P. Tuma, C. Belady, M. Fontoura, et al. Cost-efficient overclocking in immersion-cooled datacenters. In *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, pages 623–636. IEEE, 2021.

[28] N. Jones et al. How to stop data centres from gobbling up the world's electricity. *nature*, 561(7722):163–166, 2018.

[29] F. Kazhamiaka, C. Rosenberg, and S. Keshav. Tractable lithium-ion storage models for optimizing energy systems. *Energy Informatics*, 2:1–22, 2019.

[30] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[31] P. Li, J. Yang, M. A. Islam, and S. Ren. Making ai less" thirsty": Uncovering and addressing the secret water footprint of ai models. *arXiv preprint arXiv:2304.03271*, 2023.

[32] L. Lin, V. M. Zavala, and A. A. Chien. Evaluating coupling models for cloud datacenters and power grids. In *Proceedings of the Twelfth ACM International Conference on Future Energy Systems*, pages 171–184, 2021.

[33] S.-Y. Lin, W.-C. Chuang, L. Xu, S. El-Tawil, S. M. Spence, V. R. Kamat, C. C. Menassa, and J. McCormick. Framework for modeling interdependent effects in natural disasters: Application to wind engineering. *Journal of Structural Engineering*, 145(5):04019025, 2019.

[34] J. Liu. LlamaIndex. 2024. https://github.com/jerryjliu/llama_index.

[35] J. Liu, V. Hull, H. C. J. Godfray, D. Tilman, P. Gleick, H. Hoff, C. Pahl-Wostl, Z. Xu, M. G. Chung, J. Sun, et al. Nexus approaches to global sustainable development. *Nature Sustainability*, 1(9):466–476, 2018.

[36] E. Masanet, A. Shehabi, N. Lei, S. Smith, and J. Koomey. Recalibrating global data center energy-use estimates. *Science*, 367(6481):984–986, 2020.

[37] E. Masanet, A. Shehabi, N. Lei, S. Smith, and J. Koomey. Recalibrating global data center energy-use estimates. *Science*, 367(6481):984–986, 2020.

[38] D. Mytton. Data centre water consumption. *npj Clean Water*, 4(1):11, 2021.

[39] C. Nadjahi, H. Louahlia, and S. Lemasson. A review of thermal management and innovative cooling strategies for data center. *Sustainable Computing: Informatics and Systems*, 19:14–28, 2018.

[40] S.-Y. Pan, S. W. Snyder, A. I. Packman, Y. J. Lin, and P.-C. Chiang. Cooling water use in thermoelectric power generation and its associated challenges for addressing water-energy nexus. *Water-Energy Nexus*, 1(1):26–41, 2018.

[41] J. Park, T. Stavrinos, S. Peter, and T. Anderson. Empower: The case for a cloud power control plane. *HotCarbon*, 2024.

[42] Q. Pei, S. Chen, Q. Zhang, X. Zhu, F. Liu, Z. Jia, Y. Wang, and Y. Yuan. Cooledge: hotspot-relievable warm water cooling for energy-efficient edge datacenters. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 814–829, 2022.

[43] H. Ritchie and M. Roser. Our world in data. 2020. https://ourworldindata.org/energy.

[44] V. Sakalkar, V. Kontorinis, D. Landhuis, S. Li, D. De Ronde, T. Blooming, A. Ramesh, J. Kennedy, C. Malone, J. Clidaras, et al. Data center power oversubscription with a medium voltage power plane and priority-aware capping. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 497–511, 2020.

[45] A. Souza, N. Bashir, J. Murillo, W. Hanafy, Q. Liang, D. Irwin, and P. Shenoy. Ecovisor: A virtual energy system for carbon-efficient applications. In *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, pages 252–265, 2023.

[46] S. P. E. C. (SPEC). Specpower ssj2008. https://www.spec.org/power/.

[47] V. Sulzer, S. G. Marquis, R. Timms, M. Robinson, and S. J. Chapman. Python battery mathematical modelling (pybamm). *Journal of Open Research Software*, 9(1), 2021.

[48] M. Tirmazi, A. Barker, N. Deng, M. E. Haque, Z. G. Qin, S. Hand, M. Harchol-Balter, and J. Wilkes. Borg: the next generation. In *Proceedings of the fifteenth European conference on computer systems*, pages 1–14, 2020.

[49] P. Torcellini, N. Long, and R. Judkoff. Consumptive water use for us power production. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2003.

[50] P. Wiesner, I. Behnke, P. Kilian, M. Steinke, and O. Kao. Vessim: A testbed for carbon-aware applications and systems. *HotCarbon*, 2024.

[51] Q. Zhang, K. K. Ng, C. Kazer, S. Yan, J. Sedoc, and V. Liu. Mimicnet: Fast performance estimates for data center networks with machine learning. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*, pages 287–304, 2021.

[52] X. Zhu, W. Jiang, F. Liu, Q. Zhang, L. Pan, Q. Chen, and Z. Jia. Heat to power: Thermal energy harvesting and recycling for warm water-cooled datacenters. In *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, pages 405–418. IEEE, 2020.