Self-maintaining [networked] systems: The rise of datacenter robotics!

Freddie Hong
Microsoft Research
Cambridge, UK

Iason Sarantopoulos
Microsoft Research
Cambridge, UK

Elliott Hogg
Microsoft Research
Cambridge, UK

David Richardson
Microsoft Research
Cambridge, UK

Yizhong Zhang
Microsoft Research
Beijing, China

Hugh Williams
Microsoft Research
Cambridge, UK

David Sweeney
Microsoft Research
Cambridge, UK

Andromachi Chatzieleftheriou
Microsoft Research
Cambridge, UK

Antony Rowstron
Microsoft Research
Cambridge, UK

ABSTRACT

The vision of *self-maintaining systems* is to make cloud hardware automatically servicing and repairing using robotics. We define a self-maintaining system as one where software can control robotics that can automatically perform hardware maintenance tasks and repair operations. This reduces failure service windows and lowers the risk of repairs causing further cascading failures and outages. Self-maintaining systems are not purely reactive to failures, but also do proactive maintenance before failures occur which reduces future hardware failures. Operating an entire datacenter as a self-maintaining system is many years away, and we present four stages of automation, analogous to levels used for autonomous vehicles, required to reach the full vision for datacenters.

To experiment with and learn about self-maintaining systems we have focused on datacenter networking. We have created basic robots that support common network maintenance tasks, such as reseating and cleaning optical transceivers and replacing optical fiber cables. The advantages of self-maintaining networks are lower costs and increased availability and reliability. Key is a cross-layering co-design approach; the core cloud services are co-designed with the robotic systems performing the repairs and maintenance. The services control the robots, and this is very analogous to how Software Defined Networking has evolved for broader network management.

CCS CONCEPTS

• **Networks** → **Physical links**; **Network reliability**; • **Hardware**;

KEYWORDS

Self-repair, Networks, Automation

ACM Reference Format:
Freddie Hong, Iason Sarantopoulos, Elliott Hogg, David Richardson, Yizhong Zhang, Hugh Williams, David Sweeney, Andromachi Chatzieleftheriou, and Antony Rowstron. 2024. Self-maintaining [networked] systems: The rise of datacenter robotics!. In *The 23rd ACM Workshop on Hot Topics in Networks (HOTNETS '24), November 18–19, 2024, Irvine, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3696348.3696872

1 INTRODUCTION

When operating infrastructure at the scale of a large cloud provider, it is well known that hardware failures will occur frequently. In fact, so often that failures that would be considered extremely uncommon in smaller scale deployment settings will appear often at cloud infrastructure scale [13, 15].

Cloud services must remain operational despite hardware failures. These services are generally designed to run on hardware that meets or exceeds a specified peak performance level. To ensure this peak performance is consistently met, hardware is often overprovisioned to deliver higher potential performance than required. This overprovisioning might include redundant network links or spare computing and storage resources. However, this approach is costly, and cloud

HOTNETS '24, November 18–19, 2024, Irvine, CA, USA

ACM ISBN 979-8-4007-1272-2/24/11
https://doi.org/10.1145/3696348.3696872

providers typically aim to minimize overprovisioning. Alternatively, services could be designed to function on hardware with degraded performance, which requires additional design and testing efforts to ensure these services can adapt to reduced hardware capabilities. Consequently, many services prefer overprovisioning, opting for hardware that either meets the specified peak performance or is taken offline.

Modern AI clusters clearly illustrate this tension. Incorporating redundant network links between GPUs or redundant HBM memory modules at the GPU server level incurs substantial costs. These costs extend beyond financial implications to include factors such as power consumption, rack space, and chip shoreline space. This creates a dilemma: a single network link failing or an HBM module failing changes the resource availability per GPU, potentially causing significant fraction of the GPU-cluster to go offline, which is costly. However, providing a spare network link for every link in a GPU cluster, or a spare HBM module for each GPU, is simply impractical in terms of cost and energy.

To further add to the complexity, decades ago most systems designers assumed that hardware failures were fail stop. It was convenient to view a network link as either up or down, or other hardware failures like this. Unfortunately, in the cloud era many failures are not fail stop. Gray or *transient failures* are common [6]. A transient failure comes and goes, and this class of hardware failure leads to degraded performance over time with errors that appear transiently. An example would be a flapping network link, a link that oscillates between periods of normal operation and periods that exhibit high packet loss rates. Layers in the network stack will ensure retransmission of lost packets, the curse of a flapping link is the associated increase in tail latency for the network.

Often transient failures are a function of the workload or external factors, such as environmental changes in temperature, vibration and so forth. A great example would be dirt on an end-face of an optical fiber cable in a network transceiver [21]. This dirt can cause the link to fail or to flap depending on what constitutes the dirt. Transient failures can even be caused by physical technician activity in the datacenter close to the physical vicinity of the component. For example, when technicians move fiber optical cables to reach a component, the movement of the cables can cause transient packet loss in the touched cables. We refer to this phenomenon as simply *cascading failures*. Cascading failures occur when physical motion near or with hardware creates vibrations and other physical effects on the co-located hardware, which leads to additional transient (or permanent!) failures.

Regardless of the failure type, the failure cause, the degree of overprovisioning, and software services ability to operate with degraded hardware, the hardware will still need to be repaired. Physical repair is a labor-intensive task, which is performed mostly manually by humans in most datacenters. The services produce service tickets that describe what needs to be repaired or replaced and its location, and a skilled technician is assigned to perform the task. Usually, a physical repair is on a timescale of days, with a fraction of repairs being high priority and done in hours.

Something has to change, and we propose the concept of *self-maintaining systems*. A self-maintaining system is one that can manage and control its own hardware repair and maintenance. This is enabled through advanced robotics and automation. It offers the potential for fine-grained control of repairs, not only reducing the time window for a repair, but also helping manage the impact of *cascading failures* and false positives on repairs. An additional advantage is that currently very little datacenter hardware is proactively serviced, it is usually accessed only when it fails. This is due to scale (and therefore costs) and the issue of cascading failures. We believe dextrous advanced robotics design specifically to operate in the datacenter can also make proactive maintenance feasible, and thereby reduce the number of hardware failures.

There is a high amount of hardware diversity in cloud datacenters across the networking, storage and compute infrastructure. To allow us to explore the concept of self-maintaining systems we have initially been focusing on the network hardware. We believe there are early scenarios that make this the right area to focus on especially as the costs of redundancy at the network level can be high and failures are common. Failures are common due to the sheer number of network links and the nature of networking hardware means a significant fraction of hardware failures are experienced in the network. Interestingly, the failures also frequently require multiple attempts to fix due to their nature and hard to pin point the cause of errors. It should also be noted that the network has some of the highest levels of physical component diversity.

We envisage that, rather than a small number of large robots (e.g., humanoids), there will be many small robotic units that will need to collaborate to achieve network repair and maintenance tasks. We have been creating a number of small robotic units as building blocks to tackle the more common repairs. This is the first stage in enabling us to learn how to establish a self-maintaining network.

2 SELF-MAINTAINING SYSTEMS

Repairing and maintaining hardware in cloud datacenters needs to become more automated. To achieve this, it needs to be considered part of the design of any system deployed. Thinking how and when maintenance is performed, optimizing its timing, and automating and controlling the process is essential. We propose considering hardware maintenance as the lowest layer of the stack and use cross-layer communication and control to help it function. Advanced dexterous robotics capable of performing intricate hardware repairs controlled by a service API is required that allows higher layers to interact with and finely control when and how maintenance occurs. The API needs to mask the complexity but enable complex control. This is widely done for other areas, with examples including components like smartNICs and switches, and increasingly with silicon, such as TPUs, ML accelerators, and Data Processing Units. The whole approach is analogous to Software Defined Networking, and more recently to the way that power has become a first-class resource in the datacenter. Control planes manage power consumption and take actions to manage (peak) power which requires better control of component power use, and more information sharing between stack layers [18].

This requires us to understand how to *automate* hardware maintenance actions. Currently, services generate tickets to trigger technicians to repair hardware upon failures. Today's services are already good at detecting hardware failures. As we develop self-maintaining systems capable of servicing hardware autonomously, making the process more controllable and understood by the software service is essential. A fully self-maintaining system will not require the service to create a ticket describing a hardware failure; instead, it will schedule and monitor repair operations autonomously without requiring any technician intervention.

We believe the primary benefit of this approach is the significant reduction of the *service window* for failures, potentially shrinking the duration from hours and days to literally minutes. Tight coupling and control will help minimize *repair amplification* caused by cascading failures. Low-level repair actions can be correlated with any resulting failures and timed to minimize impact. Proactive measures can be taken, such as temporarily migrating loads from physical hardware adjacent to the hardware being repaired. For example, in networking, automation can report which network cables will be contacted *before* the maintenance occurs. This will enhance datacenter reliability, availability, and efficiency. Additionally, there is real potential for *right-provisioning* redundant hardware components, thus reducing the need for excessive overprovisioned online redundancy due to greater control over the window of vulnerability during hardware failures.

Achieving these goals necessitates innovative robotics and automation. Over the past few years, robotics has made significant advancements, with advanced manipulators, grippers, and highly dexterous robotics becoming more practical. Recent progress in computer vision has also made it easier to process the relatively complex datacenter environment. Nonetheless, transitioning from today's datacenters to self-maintaining ones will not happen in a single leap.

2.1 Automation levels

In order to help understand the phases of datacenter automation, we adapt the six-level driving automation taxonomy from the Society of Automotive Engineers [11] and propose the following five levels:

- **Level 0 - No Automation**. All tasks are performed manually by skilled technicians.
- **Level 1 - Operator Assistance**. Automated devices are used to augment human operators. Technicians are needed to operate the devices.
- **Level 2 - Partial Automation**. Specialized tasks are performed autonomously with human supervision or through teleoperation.
- **Level 3 - High Automation**. Fully autonomous end-to-end tasks are performed with limited human supervision.
- **Level 4 - Full Automation**. Every datacenter repair operation is fully autonomous without the need for human supervision.

To learn more about self-maintaining systems we are currently exploring robotics that operate between Levels 2 and 3. Level 3 will eventually enable self-maintaining systems, while Level 4 will enable potentially fully *self-maintaining datacenters*, not designed around humans but optimized for high density and energy efficiency instead. At Level 4 humans can provide oversight at the datacenter site but without needing to be physically present in the datacenter halls. Even achieving the basic robotics to support Levels 1 and 2 is challenging and this is our current focus. However, we believe that once Level 2 is achieved, getting to Levels 3 and 4 will be easier.

3 TOWARDS SELF-MAINTAINING NETWORKED SYSTEMS

In order to explore the challenges and opportunities, we focus on creating modular robots designed to support the network infrastructure. These can be deployed at the granularity of a hall or row of racks. The modular robots collaborate to repair hardware failures. An early learning is that the use of small modular robotics which reuse common parts keeps the costs of each robot lower than trying to use a single large robot (e.g. humanoid). This makes repair and maintenance of the robotics easier and supports incremental deployment. It also allows us to think about how to operate at height and in ways that a human or humanoid finds hard.

3.1 Datacenter networking

A datacenter network consists of server NICs, switches, routers, line cards, (optical) transceivers, and cables (fiber or copper). Short links of a few meters will use simple copper cables called Direct Attached Copper Cable (DAC). Medium length links can have a cable with integrated active transceivers at manufacture. This type of cable is known as an Active Electrical Cable (AEC) if using copper cables or an Active Optical Cable (AOC) if using fiber optic cables. Longer links will use separate optical transceivers and fiber cables with the cables inserted manually into the transceiver on-site.

3.2 Repair operations

A common operation when a link has failed or is transiently failing, for any type optical or electrical transceiver, is reseating the transceiver. Reseating simply involves removing the transceiver from the socket it is in, waiting a few seconds, and then re-inserting it. Reseating seems to have two main effects: *(i)* to potentially help the electrical connection, as gold is not immune from oxidation and corrosion, and *(ii)* to ensure a full reboot of the transceiver. This repair process is surprisingly effective, and in our datacenters, when a network link fails or flaps the first time a ticket is created for that link, the usual first step is to reseat the transceiver.

Optical fiber cables can be either single channel (LC) or multi-channel (MPO), where several fibers are packaged in a single cable. For many links the cables and transceivers are attached permanently during manufacturer. However, for links that span further (e.g. across racks) the transceiver and optical cable is supplied separately. Both the transceiver and the optical cable will be cleaned before they are connected at assembly time. If a link has failed, and a reseating of the transceiver has not solved the problem, another ticket will be generated. This will cause a technician to perform a cleaning of the transceiver and the end-face of the optical fiber cable.

This is quite complex, especially for MPO cables, where they may be 8 or more fiber channels within one cable. The cleaning process involves cleaning each fiber channel in the cable and cleaning the inside of the transceiver. The technician needs to inspect the transceiver and the end-face of the optical cable to ensure that they are cleaned according to industry specifications. This is important because dirt in these connections is a common source of link flapping, and the full impact is often dependent on temperature, humidity, vibration etc. Hence, the flapping can occur intermittently over time. As higher network link bandwidths become common, MPO cables are becoming more common, as a single fiber is (currently) able to support 100 Gbps, so an 800 Gbps link will use 8 fibers within a single MPO cable. These are more complex to clean than the single fiber variety as they have many cores in each cable, and each core needs to be independently inspected. If the transceiver has been reseated in the past, and another ticket is generated for the same link within a time window, and the transceiver and cable are cleanable, then the next stage is to perform this cleaning process.

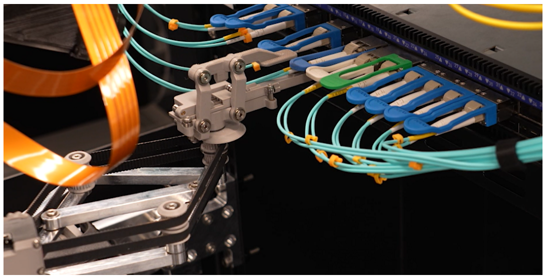

Figure 1: Prototype transceiver manipulation robot.

If the link transceivers have been reseated, and cleaned, if possible, the next common action is then to replace the transceivers and ultimately the cable. This usually requires the laying of a new fiber in trunks running beside and above the racks. If the transceiver and fiber cable are not integrated, the new installation requires the cleaning process as part of replacing the link. The replacement of an entire cable and transceivers is not trivial. If this does not solve the problem, then the final stage is to replace the NIC, line card, or switch.

3.3 Enabling a self-maintaining network

Our first goal is to develop advanced robotics that can manipulate and perform the most common tasks for optical transceivers, specifically transceiver reseating and the inspection and cleaning of both transceivers and fiber end-faces. Currently, we are not focusing on the replacement of fibers, which would involve laying new fibers. To explore the design space we have developed multiple modular robotic units.

3.3.1 Transceiver manipulation. The first robotic unit developed is a manipulator arm and gripper that allows automated transceiver manipulation, shown in Figure 1. The transceiver manipulation robot is designed to grip and manipulate a single transceiver while minimizing accidental interaction with physically close cables. This is achieved by minimizing the surface gripper area and weight. The design the of gripper has evolved over time to allow it to be inserted in between optical cables and then gently to move them apart, which still being able to grip the transceiver pull tab. Further the gripper is designed to put no pressure on the optical cable or the fiber connector, the pressure is applied to the transceiver only and only where designated by the transceiver design. The robot uses a vision system to understand the complex environment and enable it to autonomously navigate through

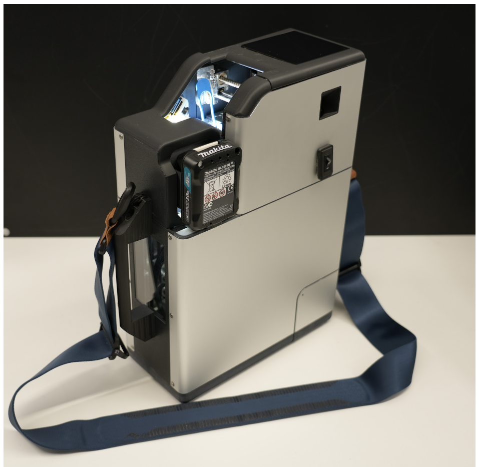

Figure 2: Prototype fiber and transceiver cleaning robot.

cluttered cabling to the target port to reseat, plug or unplug the transceiver.

3.3.2 Fiber end-face and transceiver cleaning. Figure 2 shows a fiber and transceiver cleaning robot. A transceiver with an attached fiber cable is plugged in the unit by a technician or the transceiver manipulation robot. The cleaning unit robot automatically detaches the cable from the transceiver, visually inspects the fiber end-face cores and the transceiver and then cleans any parts needed to pass inspection, before reassembling. It has many actuators, and the device is complex and dextrous. The first gripper is designed to detach the fiber cable from the transceiver (for LC and MPO cables). A core challenge is handling the transceiver and cable diversity used in a large-scale global cloud provider. The unit uses cameras and recognition models to determine the type and size of the transceiver and cable. After separation, another actuator rotates the transceiver to enable a third actuator to position the cleaning and inspection tools to clean and verify both the transceiver and cable end-face. Once this process is complete, the robot reassembles the transceiver and cable to minimize the risk of recontamination. A display mounted on the robot allows a human to monitor and observe progress, as well as see the inspected images.

The cleaning robot is modular and can be integrated with the transceiver manipulation robot and used for Level 2 or 3 automation. It can also be used by a technician as a standalone Level 1 device. When integrated with the manipulation robot, the latter handles unplugging the transceiver from the switch and inserting the transceiver into the cleaning device, and reversing the process. This entire operation currently takes a few minutes, but it can be optimized. Already, the end-face inspection for 8 cores takes less than 30 seconds which is less time than a well-trained human. The cleaning process uses wet and dry methods to address a wide range of contaminants. When the robot fails to verify the cleanliness of the transceiver or fiber it requests human support currently. Ultimately, if the transceiver is malfunctioning, the robots can carry spares, and the manipulator arm can replace the broken transceiver with a spare, and restart the cleaning process for the new transceiver.

3.3.3 Learnings. The largest challenges have been the diversity of components and high cabling density, which complicate perception and planning. The gripper design needs to support both LC and MPO cables, and some MPO cables have an 8-degree angle on the end-faces while others do not. This diversity has made handling transceivers and cables difficult. An early lesson learned was to design with more flexibility than initially required. For instance, the mechanisms that allow us to inspect MPO cables with an 8-degree end-face angle before and after cleaning can also accommodate inspections of much larger angles if needed.

The imaging inspection is free space-based with the advantage that the imaging head does not touch the cable, as opposed to the traditional tools that are used manually today. Although this approach increases complexity, it reduces fiber damage during inspection due to non-contact handling. A consequence of this is that it allows us to do detailed 3D scans of the end-face, which also opens new types of measurements (for example, detailed analysis of chips within and around the core). Now that we have got highly capable units, our next step is to really integrate them into our services software stack.

3.4 Modular robots and mobility

The units described in the previous section operate at rack scale, but for larger-scale deployment, cross-rack mobility is required. While the idea of a single humanoid robot navigating and working across the entire datacenter halls is appealing, the reality is that datacenters are designed more to tolerate humans who have to operate them rather than the other way around. Racks can be as high as 52U and removing heavy hardware items at head height and above is challenging even for a skilled human. Technicians often find it difficult to service components densely packed in racks, making the human hand less than ideal for the task. For example, the high cable density sometimes makes pulling cables to get transceivers out easier than trying to get a finger to grab a transceiver. Hence, based on our experience and observations of datacenters in operation, a gripper tailored to specific tasks can be far more efficient and can minimize cascading failures during maintenance operations. We simply do not believe that the humanoid form factor or a hand-inspired gripper is suitable for most tasks in the datacenter.

Consequently, we are currently focusing on developing a set of small-scale robotic units that minimize the variety of robot form factors needed while supporting a diverse range of operations, and this set of robotic units includes mobility units.

For these mobility units it is important to consider the operating radius for each robot. In a typical datacenter, there are several potential deployment scopes for robotics: device-level within the rack, rack-level, row-level, hall level, and full datacenter level. The chosen scope significantly influences the mobility model required and the deployment strategy. At the rack and device level, it is practical to deploy the robotics within the rack itself. For row-level mobility, the robots can potentially move along the XY plane, navigating the front or back of the racks. Finally, safety is a major concern when humans and robots need to co-exist. We are still leaning and experimenting to determine the best options.

4 DISCUSSION

One reason we selected networking to explore first is that it has the advantage that there are already complex control planes used in cloud networking infrastructure both at the datacenter [1] and WAN scale [5]. We see many parallels with Software Defined Networking, and within the community it is already common to think how to optimize the use of the network via this control plane under changing traffic or failure conditions [5, 7]. Further, it is well understood that networks (both during use and at deployment) [4, 9] can experience cascading failures due to the distributed protocols used (e.g. BGP) and nuances (and undocumented features) of their implementation on switches. This has led to both correctness checking statically before deployment (network verification) and actual emulation of switches to enable configuration checking prior to deployment [8]. We believe much of this thinking, insight and techniques will be needed to create self-maintaining systems.

In general, self-maintaining systems create a number of interesting research opportunities for the networking community:

Predictive maintenance: In today's datacenters, there is relatively little proactive work done to reduce the chance of failures. The process is mostly reactive: a service experiences what it considers a failure or needs servicing, and then generates a ticket. Proactive measures carry a real cost, including increased technician workload and potentially higher failure rates due to human error. Self-maintaining hardware can change this equation. During periods of low utilization, automation hardware can be used for proactive maintenance at little to no additional cost. For example, if several links on a switch have been fixed by reseating transceivers, the system could proactively reseat all transceivers on that switch, even if no issues have been reported. We believe this proactive maintenance could enhance reliability and availability while reducing operational costs. This also creates new opportunities to use machine learning techniques to predict failures and detect related network behavior patterns, potentially leveraging data collected by robotic systems.

Software-defined controllers: We believe an important aspect is the integration of the maintenance process into thinking about the design of core services. The ultimate goal is to increase reliability, reduce unavailability and lower service costs. An abstraction that allows a system to plan when and how to perform repairs can be very powerful. While self-maintaining hardware won't eliminate cascading failures during repairs, it will enable core services to have fine-grained control and visibility, providing real-time feedback on potential cascading errors. For example, a robot that knows when it will move cables also knows which cables and the force applied. A key challenge is designing the control plane, interfaces, and control algorithms for seamless interaction between robotic repair systems and SDN/NFV controllers. Can we develop control algorithms for automatic fault recovery and dynamic network resource reconfiguration to ensure continuous operation during repairs?

Scalable network topologies: It has been highlighted in [10] why it is important to think about the physical deployability of proposals, emphasizing the challenges of deploying many expander graph and other network topologies in the datacenter [2, 14, 17]. The reason these more efficient network topologies are not used is the complexity of deployment. At the limit, the reason why these more efficient topologies are not deployed is due to the complexity to manually deploy the complex wiring looms. We believe that we will need to think carefully about the deployability of the robots and automation that we create. In effect, we want to ensure that technicians of datacenters today do not become the technicians of robots in the future. However, if we can build self-maintaining systems, these systems may well be able to also *deploy* the network originally not just maintain it. We believe that today the arguments made about deployability limiting use of the complex networks are completely correct. However, we remain optimistic that self-maintaining systems will eventually allow for more complex and efficient network topologies (and hardware more broadly) than what is feasible today. So, in light of automated maintenance capabilities, we have the potential to design novel scalable topologies or revisit the physical deployability of previously proposed topologies that were deemed impractical to deploy in practice (e.g., expander-graph networks [2, 14, 17]). Which may work in the future, what would be needed, and perhaps we can create a metric for self-maintainability of a network design?

As an extension of this, it is interesting to explore reconfigurable network topologies to dynamically adapt to changing traffic patterns and optimize resource utilization [19]. The robotics that enables a self-maintaining network will also be able to deploy arbitrary topologies potentially. Is this useful, and if so what additional robotic functionality may we need?
Hardware redesign and standardization: Networking in datacenters appears to be the area with the most diversity. There are literally tens of different designs for optical transceivers deployed in large scale datacenters. The fundamental docking, latching, and electrical contacts are all standardized. However, the backend of the transceiver, where it is grasped by humans, can vary in color, shape, material, stiffness, and other factors. For example, each QSFP transceiver model has specified electrical interfaces and connector locations. This complexity is compounded by the increasing cable complexity, driven by the need for more bandwidth per link, with traditional methods of scaling bandwidth not keeping pace. This has led to more parallelism at the cable level, increasing the number of cores per fiber. Such diversity creates significant challenges for automation. To make self-maintenance effective, hardware should be redesigned to reduce diversity and complexity, making it easier for robots to manipulate. Investigating the redesign of hardware components to enhance their manipulability by robotic systems is a promising area for research.
Fault detection and isolation: Integrating robotics with network monitoring tools and developing algorithms for precise fault localization is another area of interest.
Energy efficiency: The community could also rethink how to enhance energy efficiency through optimized resource management facilitated by robotic systems.
Network security: An exciting area is the development of robust, integrated security frameworks and advanced monitoring systems to protect against the complex and dynamic threats introduced by robotics and automation.

5 RELATED WORK

Previous work has explored the concept of robotic patch panels through various industrial products [3, 12, 16], focusing on optical fiber switching to enhance reconfigurability. We propose using robotics to enable self-maintaining systems, rather than merely focusing on reconfigurability. When thinking of reconfigurability, there is robotics that can make the deployment and end-of-life management of racks easier. However, we see these as having a different focus to robotics designed to enable self-maintenance.

Likewise, Nokia Bell Labs has demonstrated a mobile robot capable of performing fiber switching and path verification [20]. Despite these advancements, hardware repair and maintenance in datacenters remain predominantly manual processes. Technicians are required to carry out tasks such as optical fiber cleaning and inspection physically. The high dexterity needed for these repairs, the complexity of navigating densely wired environments, and the challenges of tracking and recognizing cables, along with managing cable occlusions, continue to pose substantial difficulties for state-of-the-art robotic systems. A critical challenge in developing self-maintaining systems lies in advancing the current state of vision technology. Our work aims to address these obstacles by developing advanced perception systems capable of operating in environments characterized by complex wiring looms and significant occlusions.

6 CONCLUSIONS

We have described the concept of self-maintaining systems for the datacenter that integrate advanced modular robotics to create systems able to repair and maintain themselves automatically over time. We believe that this marks the beginning of a fundamental shift in how we conceive and design datacenter hardware and the software services. Although at the very start of this journey, and we face numerous challenges, the field feels very open and full of potential. Recent advancements in robotics suggest that the enabling technology is nearly within reach. To realize this vision it will require a highly interdisciplinary approach, with researchers in robotics and automation collaborating closely with experts in networking, systems, and machine learning.

Acknowledgements

We would like to thank the anonymous reviewers for their constructive feedback and suggestions. We would also like to thank David Bragg, Hadar Gamliel and Marco Caballero for their contributions to this work.

Disclosure of ethical concerns

This paper raises no ethical concerns, describes no research on human subjects and does not involve user data or PII.

REFERENCES

[1] Martin Casado, Michael J Freedman, Justin Pettit, Jianying Luo, Nick McKeown, and Scott Shenker. 2007. Ethane: Taking control of the enterprise. *ACM SIGCOMM computer communication review* 37, 4 (2007), 1–12.

[2] Paolo Costa, Austin Donnelly, Antony Rowstron, and Greg O'Shea. 2012. Camdoop: Exploiting in-network aggregation for big data applications. In *9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*. 29–42.

[3] Eaton Robotic Fiber Panel Systems. [n. d.]. https://www.eaton.com/us/en-us/products/backup-power-ups-surge-it-power-distribution/robotic-fiber-panel-systems.html.

[4] Ari Fogel, Stanley Fung, Luis Pedrosa, Meg Walraed-Sullivan, Ramesh Govindan, Ratul Mahajan, and Todd Millstein. 2015. A general approach to network configuration analysis. In *12th USENIX Symposium*

on Networked Systems Design and Implementation (NSDI 15). 469–483.

[5] Chi-Yao Hong, Srikanth Kandula, Ratul Mahajan, Ming Zhang, Vijay Gill, Mohan Nanduri, and Roger Wattenhofer. 2013. Achieving high utilization with software-driven WAN. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*. 15–26.

[6] Peng Huang, Chuanxiong Guo, Lidong Zhou, Jacob R Lorch, Yingnong Dang, Murali Chintalapati, and Randolph Yao. 2017. Gray failure: The achilles' heel of cloud-scale systems. In *Proceedings of the 16th Workshop on Hot Topics in Operating Systems*. 150–155.

[7] Srikanth Kandula, Ishai Menache, Roy Schwartz, and Spandana Raj Babbula. 2014. Calendaring for wide area networks. In *Proceedings of the 2014 ACM conference on SIGCOMM*. 515–526.

[8] Hongqiang Liu, Yibo Zhu, Jitu Padhye, Jiaxin Cao, Sri Tallapragada, Nuno Lopes, Andrey Rybalchenko, Guohan Lu, and Lihua Yuan. 2017. CrystalNet: Faithfully Emulating Large Production Networks. In *SOSP '17 Proceedings of the 26th Symposium on Operating Systems Principles* (sosp '17 proceedings of the 26th symposium on operating systems principles ed.). ACM, 599–613. https://www.microsoft.com/en-us/research/publication/crystalnet-faithfully-emulating-large-production-networks/

[9] Nuno Lopes and Andrey Rybalchenko. 2019. Fast BGP Simulation of Large Datacenters. In *VMCAI: Verification, Model Checking, and Abstract Interpretation*.

[10] Jeffrey C Mogul and John Wilkes. 2023. Physical Deployability Matters. In *Proceedings of the 22nd ACM Workshop on Hot Topics in Networks*. 9–17.

[11] Society of Automotive Engineers. 2014. https://www.sae.org/standards/content/j3016_202104/.

[12] OptiClarity - robot fiber optic patch panel. [n. d.]. https://opticlarity.com/products/robotics/.

[13] Bianca Schroeder, Eduardo Pinheiro, and Wolf-Dietrich Weber. 2009. DRAM errors in the wild: a large-scale field study. *ACM SIGMETRICS Performance Evaluation Review* 37, 1 (2009), 193–204.

[14] Ankit Singla, Chi-Yao Hong, Lucian Popa, and P Brighten Godfrey. 2012. Jellyfish: Networking data centers randomly. In *9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*. 225–238.

[15] Jonathan Stone and Craig Partridge. 2000. When the CRC and TCP checksum disagree. *ACM SIGCOMM computer communication review* 30, 4 (2000), 309–319.

[16] Telescent G4 Network Topology Manager. [n. d.]. https://www.telescent.com/products.

[17] Asaf Valadarsky, Gal Shahaf, Michael Dinitz, and Michael Schapira. 2016. Xpander: Towards Optimal-Performance Datacenters *(CoNEXT '16)*. Association for Computing Machinery.

[18] Jaylen Wang, Daniel S. Berger, Fiodar Kazhamiaka, Celine Irvene, Chaojie Zhang, Esha Choukse, Kali Frost, Rodrigo Fonseca, Brijesh Warrier, Chetan Bansal, Jonathan Stern, Ricardo Bianchini, and Akshitha Sriraman. 2024. Designing Cloud Servers for Lower Carbon. In *ISCA*. https://www.microsoft.com/en-us/research/publication/designing-cloud-servers-for-lower-carbon/

[19] Weiyang Wang, Moein Khazraee, Zhizhen Zhong, Manya Ghobadi, Zhihao Jia, Dheevatsa Mudigere, Ying Zhang, and Anthony Kewitsch. 2023. TOPOOPT: Co-optimizing Network Topology and Parallelization Strategy for Distributed Training Jobs. In *Proceedings of the 20th USENIX Symposium on Networked Systems Design and Implementation (NSDI)* (Boston, MA, USA). USENIX Association.

[20] Xiaonan Xu, Haoshuo Chen, Michael Scheutzow, Jesse E. Simsarian, Roland Ryf, Gin Qua, Amey Hande, Rob Dinoff, Mijail Szczerban, Mikael Mazur, Lauren Dallachiesa, Nicolas K. Fontaine, Jim Sandoz, Mike Coss, and David T. Neilson. 2023. Automated Fiber Switch with Path Verification Enabled by An AI-Powered Multi-Task Mobile Robot. In *49th European Conference on Optical Communications (ECOC 2023)*.

[21] Danyang Zhuo, Monia Ghobadi, Ratul Mahajan, Klaus-Tycho Förster, Arvind Krishnamurthy, and Thomas Anderson. 2017. Understanding and Mitigating Packet Corruption in Data Center Networks. In *Proceedings of the ACM SIGCOMM 2017 Conference* (Los Angeles, CA, USA). ACM, 362–375. https://doi.org/10.1145/3098822.3098849