

Discipline Convergence in Networked Systems

Edited by

Yungang Bao¹, Lars Eggert², Simon Peter³, and Noa Zilberman⁴

1 Chinese Academy of Sciences – Beijing, CN, baoyg@ict.ac.cn

2 NetApp Deutschland GmbH – Kirchheim, DE, lars@netapp.com

3 University of Texas – Austin, US, simon@cs.utexas.edu

4 University of Cambridge, GB, noa.zilberman@cl.cam.ac.uk

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 18261 “Discipline Convergence in Networked Systems”. This seminar explored emerging networked system design approaches, seeking to increase performance, efficiency and security through the convergence of disciplines: compute, storage and networking. With technologies such as network function virtualization (NFV) having started the convergence of computing technologies and networking technologies, this seminar discussed new research directions to embrace the convergence of disciplines that used to be mainly isolated in the past.

Seminar June 24–29, 2018 – <http://www.dagstuhl.de/18261>

2012 ACM Subject Classification Networks → Middle boxes / network appliances, Networks → Data center networks, Computer systems organization → Cloud computing, Computer systems organization → Data flow architectures, Hardware → Emerging architectures

Keywords and phrases Big data, cloud computing, computer architecture, networked systems, rackscale computers

Digital Object Identifier 10.4230/DagRep.8.6.149

Edited in cooperation with Max Plauth


1 Executive Summary

Yungang Bao (Chinese Academy of Sciences – Beijing, CN)

Lars Eggert (NetApp Deutschland GmbH – Kirchheim, DE)

Simon Peter (University of Texas – Austin, US)

Noa Zilberman (University of Cambridge, GB)

License  Creative Commons BY 3.0 Unported license
© Yungang Bao, Lars Eggert, Simon Peter, and Noa Zilberman

Networked computing systems have reached a watershed, as the amount of networked-data generated by user applications exceeds the processing capability of any single computer. This requires an integrated system design, unlike the traditional layered approaches. This seminar therefore brought together experts from the operating systems, distributed systems, computer architecture, networks, storage and databases communities, to advance the state of the art in discipline convergence in networked systems.

The networking community has advanced in giant leaps, making high bandwidth networking and software-defined networking (SDN) commodity. Furthermore, the advent of network function virtualization (NFV) has started the convergence of computing technologies and networking technologies. The computing community, on the other hand, struggled to



Except where otherwise noted, content of this report is licensed under a Creative Commons BY 3.0 Unported license

Discipline Convergence in Networked Systems, *Dagstuhl Reports*, Vol. 8, Issue 06, pp. 149–172

Editors: Yungang Bao, Lars Eggert, Simon Peter, and Noa Zilberman



DAGSTUHL
REPORTS

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

overcome power density limitations, resource- efficiency and quality-of-service etc. for cloud computing as well as end host computing (or edge computing), and cannot keep up.

Revolutionary networked system design approaches are now emerging, seeking to increase performance, efficiency and security through the convergence of disciplines: compute, storage and networking. This seminar investigated both hardware and software challenges, and attempted to bridge the gaps between different communities in order to compensate the challenges in some areas with emerging breakthroughs from other areas. Over the course of the 5-day seminar, seventeen presentations were given on various aspects of data center networking. Taking the presentations as input, the workshop then broke into five working groups to discuss research aspects of operating systems, distributed systems, computer architecture, networks, storage, and databases. The talks as well as the outcome of the breakout session and the concluding statements are summarized in this report.

2 Table of Contents

Executive Summary

Yungang Bao, Lars Eggert, Simon Peter, and Noa Zilberman 149

Overview of Talks

Hardware acceleration for Data Science as a means to discipline convergence <i>Gustavo Alonso</i>	153
The Case for Labeled von Neumann Architecture (LvNA) <i>Yungang Bao</i>	153
Distributed Join Processing on Thousands of Cores <i>Claude Barthels</i>	154
Efficient Network Communication for Data Access <i>Angelos Bilas</i>	154
A Programmable Framework for Validating Data Planes <i>Pietro Bressana and Noa Zilberman</i>	155
Compiling for Future Discipline-Converged Systems <i>Trevor Carlson</i>	155
Five Ways not to Fool Yourself <i>Tim Harris</i>	155
PASTE: A Network Programming Interface for Non-Volatile Main Memory <i>Michio Honda</i>	156
How to Make Profit: Building a Heterogeneous HPC System that Takes Advantage from Dynamic Electricity Pricing <i>Timo Hönig</i>	157
Research Directions for Edge Computing and Industrial IoT <i>Dirk Kutscher</i>	157
Efficient TCP Packet Processing <i>Simon Peter</i>	157
Enzian: A Research Computer <i>Timothy Roscoe</i>	158
The economics of consumer networks <i>Henning Schulzrinne</i>	159
Future Networks Switch Architecture <i>Golan Schzukin</i>	159
Technology and Business Challenges at Hyper-scale <i>Leendert van Doorn</i>	160
Edge to Cloud <i>Eric Van Hensbergen</i>	162
In Network Computing: Truths, Lies and Realities <i>Noa Zilberman</i>	162

Working groups

Future Systems & Disaggregated Computing
Gustavo Alonso, Trevor Carlson, Felix Eberhardt, Matthias Hille, Stefan Klauck, and Max Plauth 163

Simulation & Methodologies
Trevor Carlson, Yungang Bao, Felix Eberhardt, Stefan Klauck, Max Plauth, Golan Schzukin, and Eric Van Hensbergen 164

Edge Computing and IoT
Dilma Da Silva, Dirk Kutscher, Jörg Ott, Henning Schulzrinne, Golan Schzukin, Eric Van Hensbergen, and Noa Zilberman 165

Tools for Networked Systems
Timo Hönig, Gustavo Alonso, Claude Barthels, Angelos Bilas, Matthias Hille, Jacob Nelson, Simon Peter, Timothy Roscoe, and Leendert van Doorn 166

Hardware/Software Co-Design – Group 1 168

Hardware/Software Co-Design – Group 2 168

Hardware/Software Co-Design – Group 3 169

Panel discussions

Concluding statements
Simon Peter, Gustavo Alonso, Yungang Bao, Claude Barthels, Pietro Bressana, Trevor Carlson, Dilma Da Silva, Tim Harris, Matthias Hille, Michio Honda, Timo Hönig, Sue Moon, Jacob Nelson, Dan Ports, Timothy Roscoe, Henning Schulzrinne, Golan Schzukin, Eric Van Hensbergen, and Irene Y. Zhang 170

Participants 172

3 Overview of Talks

3.1 Hardware acceleration for Data Science as a means to discipline convergence

Gustavo Alonso (ETH Zürich, CH)

License  Creative Commons BY 3.0 Unported license
© Gustavo Alonso

Computing Systems are undergoing a multitude of interesting changes: from the platforms (cloud, appliances) to the workloads, data types, and operations (big data, machine learning). Many of these changes are driven or being tackled through innovation in hardware even to the point of having fully specialized designs for particular applications. In this talk I will review some of the most important changes happening in hardware and how they are affecting data processing. I will focus on the need to redesign the entire software stack and the opportunities offered by tailoring the stack to concrete applications. Customized stacks can be a good basis and a solid motivation for discipline convergence. As examples of the latter, I will discuss use cases from our own research that include hardware acceleration for network stacks [1, 2], in-network data processing [3, 4, 5], microservers [6, 7], and machine learning in distributed systems [8].

References

- 1 David Sidler, Zsolt István, Gustavo Alonso. Low-latency TCP/IP stack for data center applications. FPL 2016.
- 2 David Sidler, Gustavo Alonso, Michaela Blott, Kimon Karras, Kees A. Vissers, Raymond Carley. Scalable 10Gbps TCP/IP Stack Architecture for Reconfigurable Hardware. FCCM 2015.
- 3 Louis Woods, Jens Teubner, Gustavo Alonso. Complex Event Detection at Wire Speed with FPGAs. PVLDB 2010.
- 4 Zsolt István, Louis Woods, Gustavo Alonso. Histograms as a side effect of data movement for big data. SIGMOD Conference 2014.
- 5 Louis Woods, Zsolt István, Gustavo Alonso. Ibex – An Intelligent Storage Engine with Support for Advanced SQL Off-loading. PVLDB 2014.
- 6 Zsolt István, David Sidler, Gustavo Alonso. Caribou: Intelligent Distributed Storage. PVLDB 2017.
- 7 Zsolt István, David Sidler, Gustavo Alonso, Marko Vukolic. Consensus in a Box: Inexpensive Coordination in Hardware. NSDI 2016.
- 8 Mohsen Ewaida, Gustavo Alonso. Application Partitioning on FPGA Clusters: Inference over Decision Tree Ensembles FPL 2018.

3.2 The Case for Labeled von Neumann Architecture (LvNA)

Yungang Bao (Chinese Academy of Sciences – Beijing, CN)

License  Creative Commons BY 3.0 Unported license
© Yungang Bao

Conventional computer architecture usually expresses and conveys software requirements to the hardware by instruction set architecture (ISA) and virtual memory mechanism, which fail to express emerging requirements such as quality-of-service (QoS) and security. To

address this challenge, we propose a new computer architecture – Labeled von Neumann Architecture (LvNA), which enables a new hardware/software interface by introducing a hardware labeling mechanism to convey software’s semantic information such as QoS and security to the underlying hardware. In this talk, I will revisit tagged architecture initially proposed in the early 1970s. Then I will present LvNA’s principles that are different from tagged architecture and significantly reduce the design and implementation complexity. Finally I will demonstrate a RISC-V based FPGA prototype (a.k.a. Labeled RISC-V) that has been already open-sourced.

3.3 Distributed Join Processing on Thousands of Cores

Claude Barthels (ETH Zürich, CH)

License © Creative Commons BY 3.0 Unported license
© Claude Barthels

Joint work of Claude Barthels, Gustavo Alonso, Torsten Hoefler, Timo Schneider, Ingo Müller

Main reference Claude Barthels, Gustavo Alonso, Torsten Hoefler, Timo Schneider, Ingo Müller: “Distributed Join Algorithms on Thousands of Cores”, PVLDB, Vol. 10(5), pp. 517–528, 2017.

URL <http://dx.doi.org/10.14778/3055540.3055545>

Traditional database operators such as joins are relevant not only in the context of database engines but also as a building block in many computational and machine learning algorithms. With the advent of big data, there is an increasing demand for efficient join algorithms that can scale with the input data size and the available hardware resources. In this talk, we explore the implementation of distributed join algorithms in systems with several thousand cores connected by a high-throughput, low-latency network, show the impact and advantages of modern communication primitives such as RDMA, and discuss the importance of network scheduling.

3.4 Efficient Network Communication for Data Access

Angelos Bilas (FORTH – Heraklion, GR)

License © Creative Commons BY 3.0 Unported license
© Angelos Bilas

Joint work of Angelos Bilas, Pilar Gonzalez-Ferez

This talk discusses our work on how networked storage protocols over raw Ethernet can achieve low host CPU overhead and increase network link utilization for small I/O requests. We first examine the latency and overhead of a networked storage protocol directly over Ethernet and we point out the main inefficiencies. Then, we examine how storage protocols can take advantage of context switch elimination and adaptive batching to reduce CPU and network overhead. We present a system that is able to achieve $14\mu\text{s}$ host CPU overhead on both initiator and target for small networked I/Os over raw Ethernet without hardware support. Finally, I conclude with some thoughts on the role of host CPU overhead for networked I/O and its impact for fast storage devices.

References

- 1 Pilar Gonzalez-Ferez and Angelos Bilas. Reducing CPU and network overhead for small I/O requests in network storage protocols over raw Ethernet. In Proceedings of the 31st International Conference on Massive Storage Systems and Technology (MSST’2015), Santa Clara, CA, USA, June 2015.

- 2 Pilar Gonzalez-Ferez and Angelos Bilas. Mitigation of NUMA and synchronization effects in high-speed network storage over raw Ethernet. *The Journal of Supercomputing*, 72(11), 4129-4159, 2016, ISSN: 1573-0484, DOI: 10.1007/s11227-016-1726-7.
- 3 Pilar Gonzalez-Ferez and Angelos Bilas. Tyche: An efficient Ethernet-based protocol for converged networked storage. In *Proceedings of the 30th International Conference on Massive Storage Systems and Technology (MSST'2014)*, Santa Clara, CA, USA, June 2014.

3.5 A Programmable Framework for Validating Data Planes

Pietro Bressana (University of Lugano, CH)

Noa Zilberman (University of Cambridge, GB)

License © Creative Commons BY 3.0 Unported license

© Pietro Bressana and Noa Zilberman

Joint work of Pietro Bressana, Robert Soulé, Noa Zilberman

Due to the emerging trend of programmable network hardware, developers have begun to explore ways to accelerate various applications and services. As a result, there is a pressing need for new tools and techniques for debugging network devices. This talk presents NetDebug, a fully programmable hardware-software framework for validating and real-time debugging of programmable data planes. We describe validation use cases, compare our design to alternative solutions, and present a preliminary evaluation using a prototype implementation.

3.6 Compiling for Future Discipline-Converged Systems

Trevor Carlson (National University of Singapore, SG)

License © Creative Commons BY 3.0 Unported license

© Trevor Carlson

There are many issues that come up when we work to build the future datacenter that is able to reduce cost while still maintaining client SLAs. Discipline convergence, between systems and computer architecture, could provide a way forward to accomplish these goals. We propose to expose the fundamental units of potentially parallel work from the application level, as well as the system level, to the datacenter runtime. By developing applications in Domain Specific Languages (DSLs), and mapping these applications onto diverse hardware, we hope to expose potential efficiencies across the entire datacenter stack.

3.7 Five Ways not to Fool Yourself

Tim Harris (Amazon – Cambridge, GB)

License © Creative Commons BY 3.0 Unported license

© Tim Harris

URL <https://timharris.uk/misc/five-ways.pdf>

Performance experiments are often used to show that a new system performs better than an old system, and to quantify how much faster it is, or how more efficient it is in the use of some resource. Frequently, these experiments come toward the end of a project and – at

times – seem to be conducted more with the aim of selling the system rather than providing understanding of the reasons for the differences in performance or the scenarios in which similar improvements might be expected. Mistrust in published performance numbers follows from the suspicion that we optimize what we measure or that we measure what we have already optimized.

I will talk about some of the techniques I use in performance evaluation in order to try to get a better understanding of why a system behaves as it does, and why changes I make to the system lead to differences in performance. In part, the aim of these techniques is to be able to explain the performance of the system better when presenting it to other people, but the aim is also to provide me with a better understanding of the system while working on it, and to avoid me fooling myself over why some change has some particular effect.

3.8 PASTE: A Network Programming Interface for Non-Volatile Main Memory

Michio Honda (NEC Laboratories Europe – Heidelberg, DE)

License © Creative Commons BY 3.0 Unported license
© Michio Honda

Joint work of Michio Honda, Giuseppe Lettieri, Lars Eggert, Douglas Santry
Main reference Michio Honda, Giuseppe Lettieri, Lars Eggert, Douglas Santry: “PASTE: A Network Programming Interface for Non-Volatile Main Memory”, in Proc. of the 15th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2018, Renton, WA, USA, April 9-11, 2018, pp. 17–33, USENIX Association, 2018.

URL <https://www.usenix.org/conference/nsdi18/presentation/honda>

Costs of persisting data over networks have been dominated by slow access latency to disks or SSDs, and the access methods of them, causing end-to-end latency on the order of hundreds or thousands of microseconds. Therefore, networking whose RTTs over standard TCP/IP take just a few tens of microseconds, was a relatively lightweight component of the end-to-end system. However, emerging non-volatile main memory (NVMM) will change this situation, because durably writing data becomes two-three orders of magnitude faster due to not only physical media speed but also the new access methods. Therefore, network and storage stacks equally stress the end-to-end system, and tight integration of these stacks is required to design efficient systems.

We present PASTE, a new networking interface to build networked storage systems on top of it. It offers run-to-completion processing model across networking and storage layers, and true zero copy by DMA performed directly to named packet buffers on NVMM, while preserving protection and rich set of network protocols provided by the socket APIs today. We benchmark PASTE using Write-Ahead Logging and B+tree, as well as porting it to key value stores and software switch, and show PASTE significantly outperforms well-tuned Linux and the state-of-the art network stack. PASTE is an open source Linux kernel module which does not need to modify the main kernel. FreeBSD support is an ongoing effort. The work appeared in NSDI’18. [1]

References

- 1 Honda, Michio, Giuseppe Lettieri, Lars Eggert, and Douglas Santry. *PASTE: A Network Programming Interface for Non-Volatile Main Memory*. In 15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18). USENIX Association, 2018.

3.9 How to Make Profit: Building a Heterogeneous HPC System that Takes Advantage from Dynamic Electricity Pricing

Timo Hönig (Universität Erlangen-Nürnberg, DE)

License © Creative Commons BY 3.0 Unported license
© Timo Hönig

Joint work of Timo Hönig, Christopher Eibel, Adam Wagenhäuser, Maximilian Wagner, Wolfgang Schröder-Preikschat

Main reference T. Hönig, C. Eibel, A. Wagenhäuser, M. Wagner, and W. Schröder-Preikschat: “How to Make Profit: Exploiting Fluctuating Electricity Prices with Albatross, A Runtime System for Heterogeneous HPC Clusters” In Proc. of the 8th International Workshop on Runtime and Operating Systems for Supercomputers (ROSS’18), Tempe, AZ, USA, April 12, 2018, pp. 4:1–4:9, ACM, New York, NY, USA, 2018

URL <https://doi.org/10.1145/3217189.3217193>

The ongoing evolution of the power grid towards a highly dynamic supply system poses challenges as renewable energies induce new grid characteristics. The volatility of electricity sources leads to a fluctuating electricity price, which even becomes negative when excess supply occurs. This talk discusses a runtime system for heterogeneous HPC clusters that takes advantage of dynamic electricity pricing. To ensure an energy-efficient and economic processing of HPC workloads, the system exploits heterogeneity at the hardware level and considers dynamic electricity prices for runtime decisions.

3.10 Research Directions for Edge Computing and Industrial IoT

Dirk Kutscher (Huawei Technologies – München, DE)

License © Creative Commons BY 3.0 Unported license
© Dirk Kutscher

Edge- and in-Network Computing requires a rethinking of communication abstractions. This talk discusses some of the problems of current edge computing approaches with a particular focus on Industrial IoT. Instead of using client-server protocols for application-layer overlays, we discuss the possibility to conceive computing and networking holistically and propose Networked Computing Platform, providing an empowered data plane for networked computations.

3.11 Efficient TCP Packet Processing

Simon Peter (University of Texas – Austin, US)

License © Creative Commons BY 3.0 Unported license
© Simon Peter

Joint work of Simon Peter, Antoine Kaufmann, Tim Stampler, Naveen Sharma, Thomas Anderson, Arvind Krishnamurthy

TCP is widely used for client-server communication in modern data centers and across the Internet. But TCP packet handling is notoriously CPU intensive, accounting for an increasing fraction of server processing time. Techniques such as TCP segment offload, kernel bypass, and RDMA are of limited benefit for the typical small, frequent RPCs. These techniques can also compromise protocol agility, resource isolation, overall system reliability, and complicate multi-tenancy.

In this talk, I propose a unique refactoring of TCP functionality that splits processing between a streamlined fast path for common operations, and an out-of-band slow path. Protocol processing executes in the kernel on dedicated cores that enforce correctness and resource isolation. Applications asynchronously communicate with the kernel through event queues, improving parallelism and cache utilization. I show that the approach can increase RPC throughput by up to 4.1x compared to Linux. The fast-path can be offloaded to a programmable NIC to further improve performance and minimize CPU time for network processing. With hardware offload, data packets are delivered directly from application to application, while the NIC and kernel cooperate to enforce correctness and resource isolation. I show that hardware offload can increase per-core packet throughput by 10.7x compared to the Linux kernel TCP implementation.

References

- 1 Antoine Kaufmann, Simon Peter, Naveen Kr. Sharma, Thomas Anderson, and Arvind Krishnamurthy. High Performance Packet Processing with FlexNIC. In *21st International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Atlanta, GA, USA, April 2016.
- 2 Antoine Kaufmann, Simon Peter, Thomas Anderson, and Arvind Krishnamurthy. FlexNIC: Rethinking Network DMA. In *15th USENIX Workshop on Hot Topics in Operating Systems (HotOS)*, Kartause Ittingen, Switzerland, May 2015.

3.12 Enzian: A Research Computer

Timothy Roscoe (ETH Zürich, CH)

License © Creative Commons BY 3.0 Unported license
© Timothy Roscoe

Joint work of Reto Achermann, Gustavo Alonso, David Cock, Tobias Grosser, Zsolt Istvan, Amit Kulkarni, Muhsen Owaida, Timothy Roscoe, Zeke Wang

URL <http://www.enzian.systems/>

Academic research in rack-scale and datacenter computing today is hamstrung by lack of hardware. Cloud providers and hardware vendors build custom accelerators, interconnects, and networks for commercially important workloads, but university researchers are stuck with commodity, off-the-shelf parts.

Enzian is a research computer being developed at ETH Zurich (in collaboration with Cavium and Xilinx) which addresses this problem. An Enzian board consists of a server-class ARMv8 SoC tightly coupled and coherent with a large FPGA (eliminating PCIe), with about 0.5 TB DDR4 and nearly 500 Gb/s of network I/O either to the CPU (over Ethernet) or directly to the FPGA (potentially over custom protocols). Enzian runs both Barrelfish and Linux operating systems. Many Enzian boards can be connected in a rack-scale machine (either with or without a discrete switch) and the design is intended to allow many different research use-cases: zero-overhead run-time verification of software invariants, novel interconnect protocols for remote memory access, hardware enforcement of access control in a large machine, high-performance streaming analytics using a combination of software and configurable hardware, and much more.

By providing a powerful and flexible platform for computer systems research, Enzian aims to enable more relevant and far-reaching work on future compute platforms.

3.13 The economics of consumer networks

Henning Schulzrinne (Columbia University – New York, US)

License © Creative Commons BY 3.0 Unported license
© Henning Schulzrinne

Most of the key networking protocols and concepts have not changed in the past twenty years. Introductory textbooks written in around 2000 contain basically the same material as today's editions, with only concepts like ATM, X.25 and frame relay disappearing. Similarly, the IETF working group structure from 1992 resembles that of today. On the other hand, browsers, data centers and access networks have acquired new functionality, have emerged as a new area of study and engineering and have dramatically increased speed, respectively. The primary reason is that each of these other domains is dominated by a handful of vendors, at most, and typically makes it easy to maintain backward compatibility. In wide area networking, networks never die, they just slowly slowly drop nodes, with SS7 and fax still having significant commercial value.

Thus, we now have three key industries – rapidly growing data centers, stagnant access networks and disappearing “tier-1” interconnection networks, as transit prices have collapsed. At least in the US, almost all carriers are now trying to acquire businesses other than telecommunications, whether advertising or content. A lot of the Internet traffic has moved from commercial “common carrier” networks to private networks, operated by large CDNs or the hyperscale cloud providers.

For 20 years, carriers have fretted about becoming bit pipes. Initially, they were trying to add value to transport, but repeatedly struggled to make this commercially viable. This realization probably motivated their acquisition of content and advertising.

One of the fundamental challenges of the telecommunications industry is that while users value different kinds of bits, e.g., SMS or video, quite differently, possibly by orders of magnitude, they strongly resist having carriers impose content-based surcharges, particularly as the technical need for differentiation disappeared.

Most of the cost of running networks is management, not hardware, with roughly only 15% attributable to capital expenditures and a large fraction of that spent on civil engineering. Thus, carriers are primarily concerned about automation and reducing staff.

3.14 Future Networks Switch Architecture

Golan Schzukin (Broadcom – Yakum, IL)

License © Creative Commons BY 3.0 Unported license
© Golan Schzukin

Traditional switches are facing a big challenge of how to keep doubling performance each generation within a reasonable power and cost. Failing to do that will have significant impact on networks cost and power, increasing the number of tiers.

The alternative suggested and already applied in Broadcom DNX fabric switches is to distribute complexity to the edges (TOR/NIC) and keep the fabric switch simple, allowing cost and power improvements.

The simplification includes among others Packet Processing, Traffic Management, Buffering, E2E congestion control and scheduling.

3.15 Technology and Business Challenges at Hyper-scale

Leendert van Doorn (Microsoft Corporation – Redmond, US)

License  Creative Commons BY 3.0 Unported license
© Leendert van Doorn

Technology and Business Challenges at Hyper-scale

Running a datacenter at scale is all about economics. Like any other business the objective of a Hyperscaler is to increase its scale and increase margins. That means we are constantly on the look out for technologies that help us grown, reduce cost and increase the value of the services that we deliver.

However, there are two opposing forces at work in the datacenter. The cost of the infrastructure is rising faster than the price for the billable goods. So how do we deal with this? We continuously cost optimize our platform designs. We also try to get broad adoption of our platforms by open sourcing our designs within the Open Compute Platform (OCP) project. We utilize our resources more efficiently by increasing the number of VMs per node, use memory/SSD's more efficiently, etc. This goes hand in hand with more flexible customer service level agreements. We are also providing higher-level services, such as data processing and machine learning that provide higher value to our customers.

In general, we try to take advantage of commodity hardware, but we will differentiate the platforms with accelerators where we can provide value.

In this talk I gave an overview of some meta-trends, followed by specific examples for hardware and converged networking trends.

Meta Trends

There are a few meta trends that common to most Hyperscalers and that influence their decision making:

1. At the scale and growth we operate at, we cannot be beholden to a single supplier. If a supplier cannot deliver (for whatever reason), we cannot grow. So, we are actively exploring and enabling different suppliers. This applies to CPUs (Intel, AMD, ARM), networking, flash suppliers, memory suppliers, etc.
2. Moore's law is economically dead beyond 5nm. That doesn't mean that silicon technology won't scale, it does mean it is getting more and more expensive to do so at limited power and performance returns. While this trend is still a few years out, it does put a much greater emphasis on born in the cloud applications that take advantage of scale instead of single thread performance.
3. Another way to increase the performance and reduce the power is by taking advantage of application specific accelerators. These include GPGPU's, ML, FPGA's, ASICs, etc.
4. DRAM doesn't really scale anymore while we are putting more and more memory into a node to increase the VM density. So effectively the platform cost keeps increasing. Storage class memory may provide an interesting solution here since it has (long-term) a lower cost than DRAM and it is byte addressable but has orders of higher latency and a durability problem that needs to be overcome.
5. Disaggregation. We are still building datacenters with individual nodes while logically we think of, and manage, them as clusters/rows. Clearly there are optimization opportunities here by looking at clusters/rows more holistically.

6. Security is top of mind with all Hyperscalers. Obviously, platform security hygiene (such as secure boot) is critical, but also are issues such as secure supply chain and most recently isolation guarantees and side channel protection. If Meltdown and Specter have thought us anything, a mono culture is not a good place to be.

Hardware Technology Trends

There are a few hardware trends that impact our decision making. Rather than going into detail what we do specially, I'm outlining some options.

1. Networking: Every Hyperscaler has some custom software defined networking (SDN) solution. This continues to be a fruitful area of innovation, especially with the convergence of SDN, virtual switch stack, control plane, RDMA, and storage. A perfect example of this development is what AWS did with Nitro (and Mellanox with Bluefield and Broadcom with Stingray). Nitro enables AWS to offload the “expensive” x86 processor by running the bookkeeping functions on a dedicated SOC thereby increasing the number of VM's they can run on the x86 processors (apparently giving them 15% additional capacity per Nitro enabled node).
2. Offloading: The CPU is an expensive resource, so like the AWS Nitro example above, should you move functions that are “data center tax” into an SOC? What about often used data movement functions such as compression and crypto should they be move into an ASIC? What about ML inference engines?
3. Storage Class Memory (SCM): As mentioned earlier, DRAM has hit a cost plateau. Can SCM replace part of the function of traditional DRAM memory?
4. Flash: All hot storage is flash memory today (cold storage uses mechanical disks and even tape). Does it make sense to rethink the way an OS uses storage in its virtual memory infrastructure and couple it much more tightly? Especially with the advent of storage class memory? That said, the cost dynamics between flash and SCM are dramatically different so I think they'll co-exist for a long time. Of course, this is back to future and its worth to revisit some of the Multics ideas.
5. Service Level Agreement (SLA). While not strictly a hardware trend, a lot of design tradeoffs throughout the entire system are directly attributable to SLA parameters. By changing these parameters you'll get more design freedom and potentially provide more value to your customers.

Converged Networking

In the absence of the clear definition of converged networking, let me provide my own perspective on this topic. From a hyperscaler perspective there are only two interfaces into a node: networking and storage and these two are converging into a single widget. AWS Nitro is a perfect example of this convergence where an SOC provides both a networking interface and a storage interface (NVME) to the host and the guest VMs. The storage is disaggregated and in the AWS case it resides in the row (AWS EBS) although it could be distributed among the nodes or even on a remote block store depending on latency and bandwidth requirements.

With the integration of the control plane into the Smart/Secure NIC, the NIC is becoming the demarcation point between the node and the rest of the infrastructure. For all intent and purpose, it controls the node (reboot, reconfiguration, etc.), provide storage and networking services for bare metal and guest VMs.

In addition, there is a lot of interesting work going on around new types of cheaper and faster interconnects, especially at the T0 level. Examples of this are Gen-Z, new optical interfaces and increased speeds of 400Gbps and beyond.

3.16 Edge to Cloud

Eric Van Hensbergen (ARM – Austin, US)

License © Creative Commons BY 3.0 Unported license
© Eric Van Hensbergen

Joint work of Eric Van Hensbergen, Luis Pena, Geoffrey Blake, Chris Adeniyi-Jones, Girish Birajdar, Edmund Grimely-Evans

Arm is currently researching how to deal with the oncoming glut of data from a trillion Internet of Things (IoT) devices. We are approaching this via a converged infrastructure deploying cloud-like infrastructure for function-as-a-service (FaaS) between edge devices and the cloud for aggregation, analysis, filtering and regional pub/sub mechanisms. We are tackling the challenges in such an approach through hardware/software co-design, leveraging hardware acceleration to minimize overhead in the networking, memory, and storage subsystems while also looking for ways to streamline provisioning, allocation, and dispatch within the resulting distributed system. This has led us to looking at different tradeoffs in existing Operating System abstractions (in particular those dealing with accelerators) and existing microarchitectures which have evolved principally for time-sharing cores versus event-oriented processing dominating today's datacenters. In this talk, I lay out our overall vision for how such systems should work, results from our initial prototypes, and discuss opportunities and challenges for such an approach.

3.17 In Network Computing: Truths, Lies and Realities

Noa Zilberman (University of Cambridge, GB)

License © Creative Commons BY 3.0 Unported license
© Noa Zilberman

Joint work of Noa Zilberman, Yuta Tokusashi, Robert Soule, Tu Dang, Pietro Bressana, Han Wang, Ki Suh Lee, Hakim Weatherspoon, Marco Canini, Fernando Pedone, Nik Sultana, Salvator Galea, David Greaves, Paolo Costa, Peter Pietzuch, Andrew W. Moore

In-network computing enables services traditionally running on servers to run within network devices, providing orders-of-magnitude performance improvements. Still, network operators remain skeptical of In-network computing. In this talk I survey several In-network computing design efforts and discuss design trade-offs and their effect on performance, power and feasibility.

References

- 1 H. T. Dang, P. Bressana, H. Wang, K. S. Lee, H. Weatherspoon, M. Canini, N. Zilberman, F. Pedone, and R. Soulé. P4xos: Consensus as a network service. Research Report 2018-01, USI, May 2018.
- 2 N. Sultana, S. Galea, D. Greaves, M. Wojcik, J. Shipton, R. Clegg, L. Mai, P. Bressana, R. Soulé, R. Mortier, P. Costa, P. Pietzuch, J. Crowcroft, A. W. Moore, and N. Zilberman. Emu: Rapid Prototyping of Networking Services. USENIX ATC, July 2017.
- 3 Y. Tokusashi, H. Matsutani, and N. Zilberman LaKe: An energy efficient, low latency, accelerated key-value store. CoRR abs/1805.11344, May 2018

4 Working groups

4.1 Future Systems & Disaggregated Computing

Gustavo Alonso (ETH Zürich, CH), Trevor Carlson (National University of Singapore, SG), Felix Eberhardt (Hasso-Plattner-Institut – Potsdam, DE), Matthias Hille (TU Dresden, DE), Stefan Klauck (Hasso-Plattner-Institut – Potsdam, DE), and Max Plauth (Hasso-Plattner-Institut – Potsdam, DE)

License © Creative Commons BY 3.0 Unported license
© Gustavo Alonso, Trevor Carlson, Felix Eberhardt, Matthias Hille, Stefan Klauck, and Max Plauth

Key Observations:

- Hardware is changing quickly
- Software is changing much more slowly (if at all) because it is expensive
 - Because cloud software is new (and lacks significant legacy code), there is a chance to take advantage of these systems
 - Both development and maintenance are costly

Location of innovation:

- On-premises
 - Legacy systems using monolithic storage and compute
 - Hardware acceleration is enabled through commercial appliances
- Cloud-based
 - New applications are built to be stateless
 - Using higher-level constructs (Amazon Lambda, TensorFlow, etc.)
 - Not necessarily hardware optimized
 - Data-flow / event driven software
 - Requirements for Cloud-based infrastructure:
 - * A critical mass of applications and users are needed to enable cloud infrastructure

Vision Statement: A disaggregated computer system, composed of interconnected compute elements (xPU (CPU, GPU, etc.) + NIC), and collected together into a Domain Specific Architecture. Given the Hardware, we could also target the high-level stack (software) using Domain Specific Languages that would map Amazon Lambda-like tasks to hardware implementations. Not all Hardware is treated equally – but, maybe they should be. CPUs have a traditional OS stack, but GPUs, FPGAs, TPUs and xPUs do not.

Open Questions:

- Do we need a system stack (Operating System) on all accelerators (making them 1st-class, standalone citizens)?
- Does each 1st-class citizen require a network connection?
- What is the role of the network?
- How important are resilience and security?

Proposal: Disaggregated 1st-class citizens on the network with storage separated from the computation. Software is constructed with dataflow in mind, given a Domain Specific Construction (DSC) of the Virtual System and Domain Specific Languages (DSLs) to describe the computation in the system.

Aspects to Investigate:

- Description and construction of the system architecture
- Show the viability of the disaggregated machine
- Build a Virtualization Driver (VD) that controls and partitions access to hardware
- Build a Virtualization Controller (VC) that brings together (connects) VDs together into a Virtual Machine (VM)
- Investigate the role of the network (NIC + Switch)
 - Simplification of the network inside the datacenter?
- Hardware aspects to investigate
 - Multi-tenant solutions
 - Hardware threads / context management
 - Context switching
 - Performance isolation
 - Security isolation
 - Virtualization
- Software aspects
 - Support legacy software on the Domain Specific Architectures
 - Provide DSLs to build and connect tasks (like Amazon Lambda)
 - * DSLs provide optimization of software to the hardware provided
 - * Compilers target the most optimal solution, but can also target non-optimal matches as needed

4.2 Simulation & Methodologies

Trevor Carlson (National University of Singapore, SG), Yungang Bao (Chinese Academy of Sciences – Beijing, CN), Felix Eberhardt (Hasso-Plattner-Institut – Potsdam, DE), Stefan Klauck (Hasso-Plattner-Institut – Potsdam, DE), Max Plauth (Hasso-Plattner-Institut – Potsdam, DE), Golan Schzukin (Broadcom – Yakum, IL), and Eric Van Hensbergen (ARM – Austin, US)

License © Creative Commons BY 3.0 Unported license
 © Trevor Carlson, Yungang Bao, Felix Eberhardt, Stefan Klauck, Max Plauth, Golan Schzukin, and Eric Van Hensbergen

Performance Counters:

- Distributed Systems Performance Counters – Does this exist
- NICs and CPUs and GPUs have performance counters
- Datacenter Level: Operations vs. Architecture (different goals, potentially)
- Sampling device counters, new metrics for
 - SFLOW / NetFlow – Statistical Probability – Sampling vs. Tracing
 - SNMP / MIB – Management Information – Doesn't Scale (?)
 - SmartNIC / Microsoft – Datacenters might be different from traditional networking
 - * Which flows are susceptible to drops (to see if you are provisioning correctly)
- Performance Debugging
 - Standardization in collection and aggregation and

Metrics

- What are the important metrics
- Link utilization

- Tail latency (vs. average, min, max, per flow)
- Metrics on a per-area basis
- Tail latency / (Utilization * Throughput)
 - Is Tail Latency a QoS issue?
- Retransmission
- Utilization = Theoretical Peak in requests / second (memcached)
- Picking the metrics that are important – SLA is the only thing that is important (?)
- Can your cost of the components factor into how our costs are affected
- Negative Test – Invalid inputs (out-of-specification)
 - Can provide some insights into the bottlenecks and issues
 - Sensitivity Studies – Shows the weak spot in the system – For testing the robustness of the system
 - Orthogonal to CPI-stacks (computer architecture)
- LOGP analysis – can be useful for parallel computing.

Simulation

- Modeling
- Sampling
 - Multi-threaded sampling is difficult
 - There are some solutions (BarrierPoint), but they are not yet sufficient for large workloads
- Simulation
 - Parallel Simulation can be interesting – snipersim.org
 - Faster simulation solutions exist (zsim), but are not as flexible.
 - Multi-level simulation + sampling

4.3 Edge Computing and IoT

Dilma Da Silva (Texas A&M University – College Station, US), Dirk Kutscher (Huawei Technologies – München, DE), Jörg Ott (TU München, DE), Henning Schulzrinne (Columbia University – New York, US), Golan Schzukin (Broadcom – Yakum, IL), Eric Van Hensbergen (ARM – Austin, US), and Noa Zilberman (University of Cambridge, GB)

License © Creative Commons BY 3.0 Unported license
 © Dilma Da Silva, Dirk Kutscher, Jörg Ott, Henning Schulzrinne, Golan Schzukin, Eric Van Hensbergen, and Noa Zilberman

What is it? Who wants it? How can value be derived?

- It is not about having proprietary systems with control on tenancy and ownership
- Sectors that may have motivating applications: healthcare, automotive, industrial automation. smart cities, smart farms
 - More traditional enterprise applications such as management of automotive fleet
- Motivation
 - Latency
 - Bandwidth
 - Privacy
 - Services that leverage local data
 - * Data ownership is all with providers; may enable local competitors
 - * Analogy of local grocery store versus a global supermarket.

- Two possible views of the edge:
 - Carrier’s pursuit for new opportunities:
 - * Shared tenancy, general computing environment
 - * Different countries may have different models (single versus multiple carriers)
 - Home gateway services (e.g. Comcast IoT gateway)
- For IoT applications: aggregation and local processing functionality
 - Still storing in the cloud for remote access or wider aggregation
- Service provider may want to provide free service in exchange for access to the data

What are the relevant / interesting problems?

- Identify the appropriate platform(s) and infrastructure
 - Go beyond the current stack of Xeon CPUs
 - Unique platform or diversity?
- Sharing of data and computing among different vendors/providers
- Security and serviceability much harder in a distributed model
- Runtime model
 - Event-based, serverless, micro-services instead of VMs?
 - How to advertise services to the devices, relocation. Is there a central point for fallback?
 - Implications on the type of off-load supported by networking infrastructure?
- Can we have same tools/components in the cloud and at the edge?
 - Different constraints for resource management (e.g. power, thermal constraints)
 - Exploitation of temporal/spatial locality
- How to design software to run on an edge platform
 - What is the impact of doing IoT development easier or harder?
 - How does the development environment differ from existing tools for developing cloud applications?
 - Is it going to be the App Store model?
 - * Do we need globally unique IDs?
 - * From login-based to identify-based model
 - Need for fine-grain data authorization schemes (split model for sharing data, e.g., some part can go to X and another part to Y)
 - Is edge computing reinventing services that work well e.g. streaming?

4.4 Tools for Networked Systems

Timo Hönig (Universität Erlangen-Nürnberg, DE), Gustavo Alonso (ETH Zürich, CH), Claude Barthels (ETH Zürich, CH), Angelos Bilas (FORTH – Heraklion, GR), Matthias Hille (TU Dresden, DE), Jacob Nelson (Microsoft Research – Redmond, US), Simon Peter (University of Texas – Austin, US), Timothy Roscoe (ETH Zürich, CH), and Leendert van Doorn (Microsoft Corporation – Redmond, US)

License © Creative Commons BY 3.0 Unported license
 © Timo Hönig, Gustavo Alonso, Claude Barthels, Angelos Bilas, Matthias Hille, Jacob Nelson, Simon Peter, Timothy Roscoe, and Leendert van Doorn

Definition: Converged System

- Converged system must contain something additional, not necessarily a FPGA
- Disaggregated system, no straight line from compute to memory, each system component has access to the network

Simulation

- Q: Why do people simulate? Is it that people do not have systems of the size they consider?
- A: Systematically explore specific parts of the solution space. Building one thing represents a single point in the solution space. Simulating one thing yields a point in the solution space and the area around it.
- Things become problematic if model of simulator becomes too simple.
- No simulations presented during the solution talks, instead "toy systems" were used. Aren't we able to build realistic simulators?
- Explore and understand small system, then scale and go to bigger systems.
- Backlash on simulations: simulations of large scale system are unrealistic. You can't build a system that scales to the size of the planet if you only use simulation. Advice: do simulation /and/ a verification of the simulation.
- Problem in the systems community: no simulation at all, as second part of the message (verify simulated results) got lost and simulation-only work is not appreciated in the community.
- Recognize that you can get different things from different simulations; get first order information of a simulation for a reasonable design.
- Complexity is a problem for simulations; although we are not doing as complex simulations as nuclear reaction experiments people realize that experiments and simulations with cloud infrastructure is complex and thus, expensive.
- We don't invest enough intellectual efforts to build good simulation.
- Systems people still use their laptops and PCs for simulations, why not using a system at the scale of a supercomputer?

Applied Simulations

- Q: What are useful tools used at company X?
- A: Snapshotting systems, analyze them from high-level down to RPC; people often do not know what data to capture (e.g. changes in soldering process during manufacturing process that leads to disc failures in racks).
- Q: Does company X share analysis data with research?
- A: No, due to concerns wrt. customer data
- Although hyper-scalers are important, their data is not necessarily required for mid-sized systems; data can be gathered from other systems, too.
- Hyper-scaler problems often are not generic and do not apply to medium size data center.
- Deployment of new systems: build racks, 1000-10000 machines, testing first by putting them into production step by step.

Limitations

- Simulation is expensive in financial and temporal terms.
- To simulate a 500 cores system: 100us real time requires 4 days of simulation.
- Processor simulations would require 24 hours to boot Windows.
- We, the computer science community, do not think big; other fields (e.g. physics) rush out to get billions out of DoE.
- Q: Why do we not think that big?
- A1: Cultural reasons.
- A2: Different in China (taping out chips on regular basis, millions of dollars each).
- Astro physics projects are spread across many groups and use huge amounts of man-power (including CS programmers) whereas CS must exploit grad students for their research

4.5 Hardware/Software Co-Design – Group 1

Different Layers of Abstraction:

- Appliance
- Library (MPI/TensorFlow)
- Low-level API (Portals)
- DSLs
- Co-processor Offload (Command block)
- ISA

Architectural Considerations:

- Synchronous vs. Asynchronous
- Streaming vs. at rest
- Pipeline / Data flow
- Static vs. Dynamic Routing
- Static vs. Dynamic Accelerator Programming

Operating Systems Considerations:

- Security (Necessary but Orthogonal?)
- Provisioning/Workflow
- Performance Isolation (QoS)
- Multi-tenancy concerns

Hardware Interfaces:

- ROCEE/RDMA
- TCP/IP
- Command Block/Active Message
- FIFO
- Coherent Memory
- MMIO
- GPIO

Programming Paradigms:

- Workflow/Dataflow DSL
- Per Accelerator Class DSL

4.6 Hardware/Software Co-Design – Group 2

Application requirements / expectations from the network:

- HPC
- Database / Data processing
- Packet processing
- ML

Network needs to have a broad set of instructions to support the workloads mentioned above.

Need for an NISA (Network Instruction Set Arch.)

- Similar concept than ARM/x86/ etc.
- Tool chains and compilers translate high-level languages to NISA
- NISA programs can be pushed into the network
- Different network implementations can implement instructions in different ways (e.g. in different locations, such as the NIC or the switch)

Performance of these modern networks needs to become more transparent

- Network needs to advertise how fast different NISA instructions are
- Tools need to use this information and provide feedback to the user

4.7 Hardware/Software Co-Design – Group 3

Starting point of the discussion: how things are done today, both in the hyperscale world and in smaller deployments; and while there may be different concerns, there are a lot of shared ones too.

Optimization of three different resources:

- Programmer effort (education, complexity)
- Provider efficiency (SKU count, etc.)
- CPU efficiency (# VMs per machine)

Traditionally, systems are thought about as layered:

- Application/OS/Hypervisor/Node/Network
- Failures were essentially independent, per machine
- Latency model was simple: within-machine (μ s), between machine (ms), the world (s)

Running things in hyperscale clouds is more complex:

- Failures are frequently correlated
- More gradations of latency
- Programs have to think about placement and other concerns to ensure their systems provide the properties they want (performance/reliability)

Perhaps we can't decompose things in such a linear way; perhaps we need to think of overlapping domains:

- Application domains
- Failure domains
- Resource allocation domains
- Security domains
- Administrative domains
- Trust domains

It is unclear what the right “unit” of processing is for in-network computation:

- Pipes/Streams are hard to process in network devices; packets are easiest
- Are “messages” even the right unit?
- If the network does paxos, what's the right unit? Is it a “message” or “round” or something else?

5 Panel discussions

5.1 Concluding statements

Simon Peter (University of Texas – Austin, US), Gustavo Alonso (ETH Zürich, CH), Yungang Bao (Chinese Academy of Sciences – Beijing, CN), Claude Barthels (ETH Zürich, CH), Pietro Bressana (University of Lugano, CH), Trevor Carlson (National University of Singapore, SG), Dilma Da Silva (Texas A&M University – College Station, US), Tim Harris (Amazon – Cambridge, GB), Matthias Hille (TU Dresden, DE), Michio Honda (NEC Laboratories Europe – Heidelberg, DE), Timo Hönig (Universität Erlangen-Nürnberg, DE), Sue Moon (KAIST – Daejeon, KR), Jacob Nelson (Microsoft Research – Redmond, US), Dan Ports (Microsoft Research – Seattle, US), Timothy Roscoe (ETH Zürich, CH), Henning Schulzrinne (Columbia University – New York, US), Golan Schzukin (Broadcom – Yakum, IL), Eric Van Hensbergen (ARM – Austin, US), and Irene Y. Zhang (Microsoft Research – Redmond, US)

License © Creative Commons BY 3.0 Unported license

© Simon Peter, Gustavo Alonso, Yungang Bao, Claude Barthels, Pietro Bressana, Trevor Carlson, Dilma Da Silva, Tim Harris, Matthias Hille, Michio Honda, Timo Hönig, Sue Moon, Jacob Nelson, Dan Ports, Timothy Roscoe, Henning Schulzrinne, Golan Schzukin, Eric Van Hensbergen, and Irene Y. Zhang

- It's all economics as usual: companies need to increase scale or margin to stay alive
- Networks are disrupting architecture, distributed systems
- Academic disciplines need to catch-up to discipline convergence
- We should have GPUs everywhere.
- Does the end of Moore's Law mean the end of hyperscalers?

- It was great to have folks from systems, networking, PL, etc.
- Let's include storage in the next Dagstuhl seminar.

- Lots of thinking for broad range of data storage
- Three areas on an interest:
 - Data encryption
 - Recovery and handling state
 - The cost of operating at large scale

- Everyone is struggling with what to put on GPUs / accelerators
- Next 3 years see concrete advancements (general computation, not network dependent)

- There's a lot more you don't know than you do know
- In network computing: some of it quite easy on the NIC, but how would you like the network to look in 10 years time?
- The partitioning between NICs and Switches in future networks
- Diverting from just focusing on the server

- The challenge of simulating at a large scale
- How do you build a set of DSL in the system while improving/maintaining SLA
- What are the right applications?

- Not clear what are the interfaces
- The order of the abstraction layers
- "Making things with the architecture"

- Security
- The current way of thinking of things in a DC is a mess
- We were used to van Neumann model or distributed systems model. Now days we do a mesh of both models.
- Too many details, mix of topics
- Need better definitions for future – need to rethink our language – otherwise will say in ad-hoc nets
- The biggest problem is that CS still works in closed communities – how do we build heterogeneous teams? Challenge for all research institutes
- The effect of the Internet – which does drive the DC. and the concerns in the Internet are different.
- What are the commonalities and differences?
- Diversity in what people call “accelerators”.
- When are accelerators applicable? What to use when? Needs more thought + share with community
- Need abstractions
- There won't a revolution but an evolution.
- Need better use cases
- How to consider non-functional properties (e.g., power)
- DCN different to HPC networking – esp. definition of interfaces
- How the network should look for DB? Where should things be implemented?
- Challenge: implementing the convergence of the systems
- Challenges for researchers: identifying the right research question for a lone researcher
- DC designed hasn't changed a lot since 2012
- New IoT programming model: missed the streams, and how they should be processed
- Past FPGA experience was bad. Possibly need to try again. . .
- There is no dream application, which is disappointing
- Opportunities: we have pocket-technologies, but not how to “communicate” between technologies / language between technologies.
- Finding good collaborators for future work
- Need to think outside the box
- Need to identify good use cases – but limit ourselves to existing use cases
- Surprising how many people care about the network
- Challenge: how to deploy solutions? Need better open source engagement
- Challenges in user experience and QoE – need solutions not just within the DC
- Computer architecture community should learn from other communities
- In comp arch 3 approaches: pipeline, caching, parallelization – can it be used by the networking community as building boxes?
- Combination of DSL and DSA
- Seen more divergence than convergence
- Getting back to the mainframe model called “the data center”
- Need a lot of specialized knowledge – will keep maintaining hyperscalers
- Lack of security

Participants

- Gustavo Alonso
ETH Zürich, CH
- Yungang Bao
Chinese Academy of Sciences –
Beijing, CN
- Claude Barthels
ETH Zürich, CH
- Angelos Bilas
FORTH – Heraklion, GR
- Pietro Bressana
University of Lugano, CH
- Trevor Carlson
National University of
Singapore, SG
- Julian Chesterfield
OnApp Ltd. – Cambridge, GB
- Dilma Da Silva
Texas A&M University –
College Station, US
- Felix Eberhardt
Hasso-Plattner-Institut –
Potsdam, DE
- Lars Eggert
NetApp Deutschland GmbH –
Kirchheim, DE
- Tim Harris
Amazon – Cambridge, GB
- David Hay
The Hebrew University of
Jerusalem, IL
- Matthias Hille
TU Dresden, DE
- Timo Hönig
Universität Erlangen-
Nürnberg, DE
- Michio Honda
NEC Laboratories Europe –
Heidelberg, DE
- Stefan Klauck
Hasso-Plattner-Institut –
Potsdam, DE
- Dirk Kutscher
Huawei Technologies –
München, DE
- Giuseppe Lettieri
University of Pisa, IT
- Sue Moon
KAIST – Daejeon, KR
- Jacob Nelson
Microsoft Research –
Redmond, US
- Jörg Ott
TU München, DE
- Simon Peter
University of Texas – Austin, US
- Max Plauth
Hasso-Plattner-Institut –
Potsdam, DE
- Dan Ports
Microsoft Research – Seattle, US
- Timothy Roscoe
ETH Zürich, CH
- Henning Schulzrinne
Columbia University –
New York, US
- Golan Schzukin
Broadcom – Yakum, IL
- Leendert van Doorn
Microsoft Corporation –
Redmond, US
- Eric Van Hensbergen
ARM – Austin, US
- Irene Y. Zhang
Microsoft Research –
Redmond, US
- Noa Zilberman
University of Cambridge, GB

