Report from Dagstuhl Seminar 22471

Towards More Flexible and Automated Communication Networks

Artur Hecker[*1], Stefan Schmid[*2], Henning Schulzrinne[*3], Lily Hügerich[†4], Sándor Laki[†5], and Iosif Salem[†6]

1 **Huawei Technologies – München, DE.** artur.hecker@huawei.com
2 **TU Berlin, DE.** schmiste@gmail.com
3 **Columbia University – New York, US.** hgs@cs.columbia.edu
4 **TU Berlin, DE.** lily@inet.tu-berlin.de
5 **Eötvös Lorand University – Budapest, HU.** lakis@inf.elte.hu
6 **TU Berlin, DE.** iosif.salem@univie.ac.at

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 22471 "Towards More Flexible and Automated Communication Networks". Communication network are becoming more and more automated, allowing to overcome human configuration errors (a frequent reason for outages) and enabling a more fine-grained control, potentially improving also efficiency. For example, the percentage of employees of Telecom companies "really touching the network" is decreasing. The goal of this seminar was to bring together experts in the field to identify and discuss the key challenges in making communication networks more autonomous. To this end, the seminar was structured around a small number of enlightning keynote talks, leaving significant time for breakout sessions and discussions, as well as socializing.

Seminar November 20–23, 2022 – http://www.dagstuhl.de/22471
2012 ACM Subject Classification Networks → Network architectures; Networks → Network protocols; Networks → Network algorithms; Software and its engineering
Keywords and phrases networking, communication technologies, automation, programmability, flexibility
Digital Object Identifier 10.4230/DagRep.12.11.96

1 Executive Summary

Artur Hecker
Stefan Schmid
Henning Schulzrinne

Viewed from the perspective of users, communication networks just work fine and have not changed much recently. Under the hood, however, they are currently undergoing substantial changes, which are partly driven by new technologies, such as software-defined networking (SDN) and network function virtualization (NFV), and partly by new requirements, such as reducing operational costs and increasing reliability. SDN and NFV enable programming the behavior of these networks on-demand through software, so that functions and services can be flexibly deployed on short time scales at suitable locations in the network. The drawback of

[*] Editor / Organizer
[†] Editorial Assistant / Collector

this enhanced flexibility is that, without proper tools, managing and operating such networks becomes more and more challenging. This complexity and the pressures to reduce costs call for largely autonomously operating – or self-driving – networks, i.e. networks with only limited manual intervention.

It is a challenge to provide robust and performant control planes and connectivity in such highly flexible and demanding networking environments, since the notions of control and data plane, by definition, are both related to the notion of the service to be provided. Hence, a network supposed to actively support services with challenging and varying requirements, will also be flexible, and this not just in terms of how it switches flows, but also in terms of the supported queuing models, protocol stacks, deployed and used network and service functions, and even in terms of its own topology. Such a flexible data plane will require an equally flexible control plane capable both of embracing new nodes with new capacities and capabilities and of re-allocating all tasks to new nodes in a shrinking network. The sharing of resources, both in capabilities and in capacities, needs to be efficiently supported not only between different services, but also between the respective control and data planes. Moreover, suitable distributed runtime scheduling algorithms are required in order to utilize and share network resources efficiently and to fulfill highly demanding requirements from certain network slices, e.g., ultra-reliable low latency communication in case of industrial networks. Also, network debugging and diagnostics need to cope with these new demands. It needs to be investigated, to what extent artificial intelligence and machine learning can be applied. This, however, is a rather new topic in the networking domain.

Consequently, the proposed seminar brought together experts from “classical” networking, distributed systems and machine learning for networks.

2 Table of Contents

3 Challenges

The seminar revolved around the main challenges in network automation and softwarization. There is a wide agreement that communication networks should increasingly operate autonomously without manual intervention. In the context of the Internet and cloud-based systems, first steps in this direction can be observed. Telecommunication infrastructures as still applied in mobile networks slowly move towards virtual network slices and service-based architectures, but do not yet exploit the potential of increased flexibility and elasticity to a full extent. To foster this development, for example, appropriate monitoring and data analysis approaches are needed, as well as methods to describe and handle the "intent", i.e. to decide, in runtime, how the network should behave in order to fulfill user and service requirements at any given moment in time, topology and protocol.

Generally, a more flexible usage of costly network infrastructures could help to radically cut the service provisioning time while keeping the total cost of ownership low – a very important issue from the network operators' point of view. This also comprises a flexible usage of network infrastructure across different tenants with data centers of different sizes and at different locations being part of this infrastructure. In some sense, this is comparable to what happened in cloud environments and its data centers during the last years – communication networks still have to emerge along this way. This especially holds for aspects closely related to the network infrastructure itself, e.g., regarding connectivity, scheduling, consistency and the like.

Besides traditional networking techniques, aspects of distributed systems become increasingly important, e.g., with respect to runtime, non-local resource scheduling. Moreover, modelling such complex networks in closed forms (e.g., queueing theory) appears to be increasingly less promising. Focusing, for example, on single TCP connections in the context of congestion control in such emerging complex and more and more automated communication environments appears to have strong limits. Therefore, machine learning attracted more attention in the networking community.

Consequently, autonomously operating and self-driving communication networks can highly profit from an interdisciplinary approach, e.g., including "classical" networking, distributed systems and machine learning.

More specifically, synergies among the following aspects of automation in communication systems are identified:

- Deployment and dynamic adjustment of (virtualized) network and upper-layer services including demand-driven relocation of functionalities and services.
- Robust and performant control planes in highly dynamic and autonomous communication systems that share common networking resource potentially also with the data plane.
- Network debugging and diagnostics (e.g., automated detection of routing failures or DDoS attacks).

These apply to an entire spectrum of networks, including wired/wireless/cellular/hybrid networks. This way, we may be able to discover new synergies and application domains. This follows also the observation that the network landscape is becoming increasingly more diverse considering both, the technological as well as the administrative domain. So this calls in general for concepts that can deal with this increasing diversity.

4 Program and Agenda

To break the ice, we started the seminar with a short round of introductions, where each participant also stated his or her goal for the seminar.

Among other, the following questions came up in this round:

- What are the goals of network automation? Saving human labor? Increasing network efficiency and reliability?
- How automatic and flexible are existing networks?
- Network automation, programmability and SDNs are no longer new – what has worked? What hasn't worked quite as well?
- What are specific missing pieces?
- What are the challenges of doing research in this area?
- What are gaps in practice and understanding?
- In which sense is current practice not "best" practice?
- Where is it still painful, why did certain ideas not work as well as we hoped?
- How should ideas be evaluated? Do they need standards or other infrastructure?

Based on these questions, we structured the breakout sessions according to the time scales, at which network adaptations need to happen:

N Network planning and design, spectrum planning [typically weeks]
C Configuration, spectrum agility, and programming (data centers, network elements, end systems) [days]
T Traffic engineering [hours]
D Fault detection and diagnostics [minutes]
F Traffic and flow management (congestion, QoS, routing) [seconds and below]

We organized three large breakout sessions, in addition to the discussions during the social program (hike to the castle and excursion):

1. Flexibility = Programmability and observability? What are the required abstractions for automation, what are the programming models (P4 and beyond)? What is the role of network management, what can be useful approaches beyond SNMP? How should configuration languages and approaches evolve?
2. What is the status quo of automation for different network types? Including cellular networks (4G, 5G, 6G, ...), enterprise, home and wired access networks (Ethernet, WLAN, PON, HFC, ...), datacenter networks and cloud (SDN, ...), non-traditional networks (e.g., NGSO satellite, mesh networks, industrial, ...)? What are the pain points and where do we need more research? How can automation benefit security, QoE, cloud configurtion, etc.?
3. How can we facilitate research, and how can we educate students in these areas? What is needed to do high-impact research? What infrastructure exists for research and teaching, in terms of testbeds, emulators, simulators? Where are gaps in the curriculum, e.g., on AI/ML, formal verification, (digital) twins, etc.

5 Discussions

5.1 Day 1 (Lily Huegerich)

During the first day, several key themes emerged with regard to the future of network automation. Among these themes were the concepts of intent-driven and self-optimizing networks as well the role of the human in the control loop.

To summarize the many components that fit into these themes we will give a description of a hypothetical system that fulfills the aforementioned qualities. It all starts with the user's intent. The user uses a high-level declarative language of what qualities they want their system to fulfill. Given this and a set of diverse hardware components and constraints, the system is capable of translating the declarative statements of intent into concrete configurations and programs for managing the user's network. The user is able to, but does not need to exercise a high degree of control over the details of the solution, and can specify unchangeable intents as well as less stringent desires for the outcome. The generated configurations and programs are designed to be optimized for the specific hardware on which they will be deployed. The system makes appropriate choices for which functions should run on which network component. These choices take into account the capabilities and limitations of the various components, as well as their relationships to other available components. This helps to ensure that the system is able to make the most effective use of the available hardware.

Here we can see an example of an advanced network automation system that is intent-driven. The user states declaratively and generally what they need and the system from this derives the technicalities. Once the system has a preliminary design the network is able to self-optimize while the network is running. Examples of self-optimizing include adapting switch code to run time traffic or adapting to changing network topologies. The system does this and more in real-time. It is also able to handle network split and rejoining depending on the current state and intent that the network must fulfill. Its self-optimization is in all levels of the network ranging from in-network micro burst management all the way to physical hardware planning. In the case of physical infrastructure, the described system is able to evaluate the feasibility and potential benefits of changes to the user's organization's physical infrastructure. This might include the assessment of the availability of necessary components such as cables, servers, and radio infrastructure, as well as personnel time resources. If a change is deemed beneficial and possible, the system can plan and coordinate every step of the process, including ordering supplies and scheduling technicians.

During this entire process, the human/user is able to remain in the loop. This is because each adaptation in the network is accompanied by an explanation for the change, the reason for it, and the specific modifications made to fulfill the identified requirement. The system as well maintains a clear version control history for the entire system's state, allowing users to revert to a previous state of configuration and programming at any time. The user retains the ability to zoom in and examine the details of the system while it is running, and the system is able to identify specific network components that may be responsible for certain aspects, upon request.

This hypothetical system encompasses many discussed goals, visions, and even existing technologies that were deemed to ease the work of network deployment and management which would bring more flexibility and automation to the field of networking. To bring the state of network management closer to a state were a system like this would be possible, there are a number of areas of research that would be necessary. Including the following:

1. The translation of declarative intent-based statements into concrete configurations and programs. This might be done with natural language processing, or potentially with a declarative programming language.
2. Automation of hardware-specific optimization.
3. Assisting in automizing the currently mainly manual process of human and physical resource management and scheduling needed changes in physical infrastructure.
4. Expand on current network self-optimization research and allow for heterogenous network components to optimize their goals jointly.

5. Find methods that balance and rank the importance of detected anomalies to prevent alarm fatigue while still detecting failures, even grey failures.
6. Expand on network visualization tools.
7. Generate a model that is able to identify the source of a certain network quality. For a simple example, the user would want the reason for packet loss in the network and the system could identify the misconfigured switch responsible.
8. Work on incorporating user-friendly version control in network configurations.

In addition to the technological goals that were discussed during the seminar, the concept of responsible networks was also discussed. This included questions as how to identify or locate the source of certain qualities in the network, such as energy consumption or privacy policy violations. The natural follow to this question is how to blame the network, or how to motivate the controllers of network components to adapt their behavior when thes!!e components are deemed responsible for undesirable behavior in the network. This discussion highlights the importance of considering not only the technical aspects of network automation but also the economic and policy factors that can influence the behavior of network actors. The realization of the goals outlined in this text has already begun in various areas. The next steps will involve the expansion of these concepts to multiple areas and for them to cooperate with one another. Lastly, to have all the components smoothly function with one another, have the quality of portability, and a pleasant user interface.

5.2 Day 2 (Iosif Salem)

In the breakout sessions of day 2, we discussed the possible future goals of network automation (NA). We argued that the attempt of categorising NA related problems according to time scale, i.e. problems relevant seconds, minutes, weeks, etc, was not straightforward. However, control loops seem like a good candidate for categorising network management in different time-scales. We then took a step back and discussed what network management (NM) should include. The general context is flexibility and automation. Specifically, NM should include provisioning, performance monitoring/management, fault diagnosis, connection to network (client to cloud/VM, one-to-one/many, bootstrapping), traffic monitoring, security, VNF, CNF, and optimization, to name a few. We also discussed the comparison of NM and SDN (e.g. difference in security requirements). NM can be viewed as a recommendation system, giving alternatives (e.g. via AI) with performance guarantees to the network administrator on how to serve the intended policy in the best way possible. To that end, there could be a hierarchy of management policies. For example, if the network management configuration at level 1 is problematic, the network can always fall back to the level 0 network management configuration (or by keeping a bounded history of snapshots).

We agreed that a main goal of network automation should be to reduce or remove the error prone human work, given that networks are becoming increasingly more complex to manage. Instead, a promising vision would be declarative (or intent-based) network management. This means that the human in the loop should be restricted to declaring the managing policy/requirements, which should be the input to automated NM. That is, humans should be the source and sink of NM, while the remainder of NM should be automated (self-driving networks). In terms of layering, we envisioned a network management layer that focuses on intent. Intent can be expressed in a domain-specific language or using NLP. Types of intent may include: reachability, application performance, security level, encrypted storage, privacy, throughput, responsible networking, etc. The MP also comes with challenges in orchestration

and providing guarantees, or even clashes of (declared) intents or with regulations. Also, a proposed network configuration (computed by the declared intent) might prioritise some customers over others, but should respect SLAs.

5.3 Day 3 (Sándor Laki)

Network intelligence can be split into two main layers: slow and fast thinking. Slow thinking requires pre-existing knowledge, more capacity and computational complexity, and expensive infrastructure. It operates on longer time scales and is responsible for high-level decision-making. In contrast, fast thinking is similar to basic models that can efficiently be executed on constrained hardware but have quick response times. This lower layer often has to make decisions at the packet level on microseconds or shorter time scales. AI-native network automation must combine fast and slow thinking to manage the different control tasks at various levels. Currently, we lack a comprehensive solution, and there are many open questions in this area. The AI models must be interpretable, faithful, and aligned with the regulations (complex law landscape like AI Act in the EU). The maintenance of machine learning models has high algorithmic complexity that is hard to handle with existing tools. Deployment of AI-native solutions is also challenging since heterogeneous hardware is located at different infrastructure locations. The efficient handling of such infrastructure complexity is still an open question, including the optimal placement of AI functions and orchestration. Model execution may have high computational complexity that leads to increased energy consumption. Methods for reducing this complexity and lightweight models with high expressiveness are needed to lower energy consumption, making AI-native approaches greener.

Evaluation of network automation methods requires realistic simulators or testbeds. An ideal testbed should be comprehensive, cover multidomain scenarios, allow testing in the wild with a full stack of various network protocols, enable the emulation of failures, outages, etc., and emulate end-to-end behaviors (even across multiple networking domains). Existing testbeds focus on specific networking aspects, and they could be more user-friendly in many cases. These environments often do not meet the users' expectations. Setting up the environment from scratch is often painful. Tenant resources or the level of access to them is limited. They need to scale better; e.g., they should be able to emulate millions of end users in multiple ISPs/ASes, etc. An ideal platform should also support deep programmability from management to data planes. Automation of the testbed itself is also needed with increased visibility and proof that infrastructure monitoring cannot affect the experiments/measurements. There are good practices in the area of service-meshes that support Canary deployment in production environments. Accordingly, the production environment is not separated from the testbed; e.g., a subset of the traffic (e.g., opt-in users) can be redirected to the new services to validate their correctness. Similar ideas may be applicable in the networking domain, e.g., at academic ISPs. In addition, automated testing of networks can also learn from other areas like chaos engineering. New methods for quantifying network test coverage are also needed for proper evaluation. The smooth deployment of new network releases also poses many challenges, including deployment automation, failure handling, and automated assessment. This may require new network verification methodologies and tools similar to CI-like approaches in traditional software development.

In addition to testbeds, data sets and simulators are also needed for evaluating new methods. Most available traces and data sets in the networking area are very old (e.g., from 1989). Using them in recent papers is not really useful. There is no joint community effort to collect available traces, data sets, and traffic generators at a public repository. If data collection in real environments is not possible, methods need to be developed to create synthetic data, enabling research reproducibility. Obtaining large network topologies is also problematic since, e.g., the largest topology in TopologyZoo only has 794 nodes. To evaluate new network automation methods, we also need to emulate failures that can happen in real-world operations. Failure models are needed for claims of generality. In general, accurate models are needed for topology, traffic, faults, and policies. However, the involvement of consumer ISPs could be required to get accurate data necessary for deriving usable models. Companies need to see the value of exporting their data. Without the involvement of ISPs, we can only see the symptoms without indicating the root cause. For example, RIPE Atlas could see network failures, but the root cause remains hidden. Emulators often introduce side effects in the measurements. These problems must be listed before using a tool or analyzing the results.

6 Conclusion

Overall, the seminar was a great experience and very productive. For many participants it was the first in-person meeting after the pandemic, which made the seminar particularly special and intensive. The many opportunities to meet also in smaller groups sparked many smaller projects, from which we expect several research activities in the next months. Given the success of the seminar and the importance of the topics, we hope we can organize another edition of the seminar in 2-3 years.

7 Overview of Talks

7.1 Self-Optimizing Network Services

Gianni Antichi (Queen Mary University of London, GB)

Main reference Sebastiano Miano, Alireza Sanaee, Fulvio Risso, Gábor Rétvári, Gianni Antichi: Domain specific run time optimization for software data planes. ASPLOS 2022: 1148-1164
URL https://dl.acm.org/doi/10.1145/3503222.3507769

State-of-the-art approaches to design, develop and optimize packet-processing programs are based on static compilation: the compiler's input is a description of the forwarding plane semantics and the output is a binary that can accommodate any control plane configuration or input traffic.

In this talk, I will demonstrate that tracking control plane actions and packet-level traffic dynamics at run time opens up new opportunities for code specialization at both software and hardware level. I will introduce a number of new techniques, from static code analysis to adaptive code instrumentation and discuss a toolbox of domain specific optimizations that are not restricted to a specific data plane framework or programming language.

7.2 Self-Adjusting Networks

Chen Avin (Ben Gurion University – Beer Sheva, IL)

Joint work of Chen Avin, Stefan Schmid
Main reference Chen Griner, Johannes Zerwas, Andreas Blenk, Manya Ghobadi, Stefan Schmid, Chen Avin: Cerberus: The Power of Choices in Datacenter Topology Design – A Throughput Perspective. Proc. ACM Meas. Anal. Comput. Syst. 5(3): 38:1-38:33 (2021)
URL https://doi.org/10.1145/3491050

Modern datacenter applications' bandwidth and latency requirements have led researchers to propose various datacenter topology designs using static, dynamic demand-oblivious (rotor), and/or dynamic demand-aware switches. However, given the diverse nature of datacenter traffic, there is little consensus about how these designs would fare against each other. In this talk, I will present the vision of self-adjusting networks: networks that are optimized toward and "match" the traffic workload they serve. We will discuss information-theoretic metrics to quantify the structure in communication traffic, the achievable performance in datacenter networks matching their demands, present network design principles accordingly, and identify open research challenges.

In addition, I will discuss Cerberus, a unified, two-layer leaf-spine optical datacenter design with three topology types. Cerberus systematically matches different traffic patterns with their most suitable topology type: e.g., latency-sensitive flows are transmitted via a static topology, all-to-all traffic via a rotor topology, and elephant flows via a demand-aware topology. We show analytically and in simulations that Cerberus can improve throughput significantly compared to alternative approaches and operate data centers at higher loads while being throughput-proportional.

7.3 KIRA – A Scalable ID-based Routing Architecture for Control Planes

Roland Bless (KIT – Karlsruher Institut für Technologie, DE)

Joint work of Roland Bless, Zoran Despotovic, Artur Hecker, Martina Zitterbart
Main reference R. Bless, M. Zitterbart, Z. Despotovic and A. Hecker, "KIRA: Distributed Scalable ID-based Routing with Fast Forwarding," 2022 IFIP Networking Conference (IFIP Networking), 2022, doi: 10.23919/IFIPNetworking55013.2022.9829816.
URL https://doi.org/10.23919/IFIPNetworking55013.2022.9829816

Future networks will become even more complex than today's networks. Even very well managed networks recently suffered from large outages due to configuration mistakes made by its operators. In some cases the operators were not able to get back to the routers, because they also lost control to their control plane.

In this talk we present KIRA, a scalable ID-based routing architecture that was specifically designed for control planes. Automation of future networks requires a scalable, zero-touch control plane fabric that interconnects all networked resources (e.g., switches, routers, servers, storage, base stations, ...) providing reliable control access to them. Since future networks are more dynamic due to increased use of virtualization and also usage of mobile nodes like drones, it is useful to have an in-band control solution that does not require a dedicated, separate out-of-band management network, which also requires its own setup, configuration, and management. Furthermore, in-band control allows to let the control plane expand everywhere the network resources come into existence.

KIRA is a scalable routing solution that is tailored to control planes, i.e., in contrast to commonly used routing protocols like OSPF, ISIS, BGP etc., it prioritizes resilient connectivity over route efficiency. It scales to 100,000s of nodes in a single network, it uses ID-based addresses, is zero-touch, and is able to work well in various network topologies. Moreover, it offers a flexible memory/stretch trade-off per node, shows fast recovery from link or node failures, and is loop-free, even during convergence. KIRA is composed of R^2/Kad in the routing tier and of a PathID-based forwarding scheme in its forwarding tier. R^2/Kad constructs underlay routes by using a Kademlia-directed ID-based overlay routing strategy and uses source routing between the overlay hops. For control plane packets source routing would induce some per-packet overhead that is avoided by replacing the source routes with PathIDs. PathIDs are similarly used as in label switching and can be partially calculated beforehand. So KIRA establishes a control plane fabric that provides zero-touch connectivity between all networked resources in a scalable manner. Control plane elements such as SDN controllers, Kubernetes controllers, Path Computation Elements, Management servers, and so on can run on top of this control plane fabric. We see such a solution as a foundation that is required for (future) network automation as it constitutes a reliable base for network control that will always work as long as there is physical connectivity available between the entities.

7.4 Low-latency Networks Slicing with Programmable Network Elements

Georg Carle (TU München, DE)

Using programmable network components may lead to unwanted performance impairments. This is of particular concern when aiming to provide performance guarantees such as worst-case latencies. In this talk, the issues of how processing architecture, operating system and system configuration, and partitioning of resources among different tenants are addressed. As main method to gain insights on these issues, a data-driven approach with systematic experiments is introduced, which raises the need of reproducibility. As already identified in the workshop on Models, Methods and Tools for Reproducible Network at SIGCOMM 2003, reproducibility is challenging in networked systems research. While significant progress meanwhile has been achieved, further effort remains needed. Flexible traffic generators like MoonGen, and experiment automation with orchestration solutions such as pos, the plain orchestration service, support automated testbed operation, automated experiment execution, and high quality experiment results. Extreme Value Theory is an approach that utilizes real-world measurement data, and that can be used to model predictions on tail latency quantiles on a flow level. This approach is promising for dimensioning networks under latency-constrained service level agreements.

7.5 Native Network Intelligence, Fast and Slow

Dario Rossi (Huawei Technologies – Boulogne-Billancourt, FR)

Joint work of Dario Rossi,Zhang Liang
Main reference Dario Rossi, Liang Zhang: "Network artificial intelligence, fast and slow", in Proc. of the 1st International Workshop on Native Network Intelligence, NativeNi 2022, Rome, Italy, 9 December 2022, pp. 14–20, ACM, 2022.
URL https://doi.org/10.1145/3565009.3569521
Main reference Dario Rossi, Liang Zhang: "Landing AI on Networks: An Equipment Vendor Viewpoint on Autonomous Driving Networks", IEEE Trans. Netw. Serv. Manag., Vol. 19(3), pp. 3670–3684, 2022.
URL https://doi.org/10.1109/TNSM.2022.3169988

As networks have historically been built around connectivity, architectural features concerning quality of service, mobility, security and privacy have been added as afterthoughts – with consequent well known architectural headaches for their later integration. Despite Artificial Intelligence (AI) is more a means to an end, that an architectural feature itself, this is not completely different from what concerns its integration: in particular, while Cloud and Edge computing paradigms made it possible to use AI techniques to relieve part of network operation, however AI is currently little more than an additional tool. This talk describes a vision of future programmable networks, where AI becomes a first class commodity: its founding principle lays around the concept of "fast and slow" [1] types of AI reasoning, each of which offers different types of AI capabilities to process network data – in particular to make use of existing knowledge, or to create and extend the knowledge itself. Introducing emerging AI-to-AI communication patterns as we move towards more intelligent networks, we also outline desirable properties (explainable automated, fit and sustainable) of AI native networks.

References

1 D. Kanheman, "Thinking, fast and slow", 2011

Participants

- Gianni Antichi
Queen Mary University of London, GB
- Chen Avin
Ben Gurion University – Beer Sheva, IL
- Roland Bless
KIT – Karlsruher Institut für Technologie, DE
- Georg Carle
TU München, DE
- Klaus-Tycho Foerster
TU Dortmund, DE
- Fabien Geyer
Airbus – München, DE
- Sergey Gorinsky
IMDEA Networks Institute – Madrid, ES
- David Hay
The Hebrew University of Jerusalem, IL
- Artur Hecker
Huawei Technologies – München, DE
- Lily Hügerich
TU Berlin, DE
- Karin Anna Hummel
Johannes Kepler Universität Linz, AT
- Sándor Laki
Eötvös Lorand University – Budapest, HU
- Shir Landau Feibish
The Open University of Israel – Raanana, IL
- Huiran Liu
TU Berlin, DE
- Gábor Rétvári
Budapest University of Technology & Economics, HU
- Dario Rossi
Huawei Technologies – Boulogne-Billancourt, FR
- Iosif Salem
TU Berlin, DE
- Gabriel Scalosub
Ben Gurion University – Beer Sheva, IL
- Stefan Schmid
TU Berlin, DE
- Henning Schulzrinne
Columbia University – New York, US
- Cigdem Sengul
Brunel University – London, GB
- Martina Zitterbart
KIT – Karlsruher Institut für Technologie, DE