

# Synthesizing Plausible Infrastructure Configurations for Evaluating Edge Computing Systems

Thomas Rausch  
*TU Wien*

Clemens Lachner  
*TU Wien*

Pantelis A. Frangoudis  
*TU Wien*

Philipp Raith  
*TU Wien*

Schahram Dustdar  
*TU Wien*

## Abstract

This paper proposes a framework for synthesizing infrastructure configurations for evaluating edge computing systems under different conditions. There are a number of tools to simulate or emulate edge systems, and while they typically provide ways of modeling infrastructure and network topologies, they lack reusable building blocks common to edge scenarios. Consequently, most edge computing systems evaluations to this date rely on either highly application-specific testbeds, or abstract scenarios and abstract infrastructure configurations. We analyze four existing or emerging edge infrastructure scenarios, from which we elicit common concepts. The scenarios serve as input to synthesize plausible infrastructure configurations, that are parameterizable in cluster density, device heterogeneity, and network topology. We demonstrate how our tool can generate synthetic infrastructure configurations for the reference scenarios, and how these configurations can be used to evaluate aspects of edge computing systems.

## 1 Introduction

It is challenging to evaluate edge computing systems, such as compute platforms [35], communication middleware [36], or resource allocation methods [41], in a way that allows generalizable conclusions. We attribute this to (a) the inherent complexity and heterogeneity of such systems, and (b) a scarcity of available reference architectures for edge infrastructure, standardized benchmarks, and data on real-world deployments. Evaluating cloud computing systems in simulated environments has become comparatively straight forward due to the large body of architectural models, benchmarks [22], and trace data sets from production-grade cloud platform deployments [48], that facilitate well grounded systems evaluations. In contrast, edge computing researchers currently rely on either highly application-specific testbeds [20, 24], or abstract architectures such as P2P or hierarchical three-tiered topologies [41, 46, 50].

We propose a framework for synthesizing plausible edge infrastructure configurations from reference scenarios, for

evaluating edge computing systems under different conditions and parameters. We examine the commonalities of emerging application scenarios that leverage edge infrastructure in the entire compute continuum, to elevate common concepts into our framework domain. The result is a set of primitives and building blocks for generating edge infrastructure configurations and topologies. Based on data we have gathered for the scenarios, we identify a parameter space for these building blocks, to give researchers the ability to vary different aspects of the generated topology, s.t. the cluster topologies remain plausible extensions of the original scenarios grounded in empirical data. We demonstrate the applicability of our open-source tool, *ether* [32], by synthesizing configurations for selected scenarios, and showing how the synthetic configurations can be used as input for a simulator to evaluate systems aspects such as data flow in a serverless edge platform [35].

## 2 Related Work

There are a number of tools to simulate or emulate edge systems [18, 19, 26, 42]. For defining topologies, these simulators either let users import topologies that were created up-front using random generators like BRITE [26, 27]; or provide APIs for constructing topologies programmatically [19], through configuration files [42], or graph formats [18]. Our framework can be used to generate plausible infrastructure configurations grounded in existing scenarios as input for these tools.

Existing approaches to generate randomized network topologies rely on abstract models such as random geometric graphs [15, 49], or hierarchical structures and power-law degree distributions [27]. Network research has been investigating Internet topologies for decades. There are several approaches to procedural topology generation, such as [9, 27, 44]. These approaches aim to be generic, and focus on Internet-scale topologies. There is no distinction of host device types, capabilities, or reusable building blocks (such as edge data centers [38], or portable edge computers [33]). Our approach is different in that it provides high-level concepts from existing edge infrastructure and hybrid edge cloud scenarios.

### 3 Edge Infrastructure Scenarios

We outline several scenarios where edge infrastructure, and heterogeneous pools of edge resources, form a distributed compute fabric to facilitate emerging applications. From the scenarios we identify common concepts that we then elevate into our framework’s domain. We also report concrete numbers on scenario-specific infrastructure we have found.

#### 3.1 Scenario Descriptions

**Scenario 1 – Urban Sensing** Urban sensing is a smart city concept, to provide citizens and governmental parties with environmental and contextual urban data. More and more cities [12, 28, 43] deploy IoT nodes with cameras and sensing capabilities into urban areas to enable a variety of applications. Data can be used for urban monitoring applications; to create data analytics models such as crowd behavior, flooding models, or accident risk prediction [8]; or to enhance human cognition [34]. An example is the Array of Things (AoT), a networked urban sensing project initiated in Chicago, which aims to provide access to real-time data on urban environments [11]. Each node is equipped with two Single Board Computer (SBC) devices, sensors, and a camera. Nodes are deployed throughout Chicago’s 77 neighborhood areas, e.g., mounted on lamp posts or mobile base stations, and networked with, e.g., cellular networks, WiFi, or wired uplinks. As of Spring 2019 the deployment comprises about 200 nodes.

**Scenario 2 – Industry 4.0** Edge computing is considered a key component of realizing Industry 4.0 concepts such as smart manufacturing or the Industrial IoT (IIoT) [13]. Consider an international manufacturing company that manages several factories. To facilitate IT/OT convergence, factory floors are equipped with edge computing hardware, IIoT gateways, SBCs as sensor aggregators for soft real-time analytics and preprocessing, embedded AI hardware for low-latency video processing (e.g. for high-speed manufacturing lines), and small form-factor computers as on-site workstations. These are plausible extensions to the prototypes presented in [13], and the general trend towards using embedded AI hardware for using sensor and video analytics in predictive maintenance scenarios [51]. On-premises edge data centers (DC) with varying resource capabilities provide additional IT infrastructure for various compute loads and services. Premises are interconnected through shared cloud resources. The infrastructure forms a federated edge-cloud environment that includes resources from across the compute continuum.

**Scenario 3 – Telco-operated mobile edge clouds** Multi-access Edge Computing (MEC) allows third parties to deploy applications on a Mobile Network Operator’s (MNO) edge DCs. MEC standardization efforts driven by ETSI include a MEC reference architecture [16]. However, MEC topologies highly depend on the operator’s dimensioning decisions, and

different candidate locations for edge DCs exist. Installing compute infrastructure directly at base stations would offer significant latency benefits, but may be hindered by space limitations, especially in urban areas. Another option is to deploy edge DCs at the MNO’s central offices (CO) [30], which aggregate traffic from base stations over high-capacity and low-latency links. This performs well in terms of latency and has advantages from a DC “real estate” perspective. A third scenario is to push MEC to core network DCs, co-locating it with the Packet Gateway (P-GW). This is technically the simplest option for legacy 4G networks [21], at the cost of increased latency and reduced traffic offloading gains.

Data of real-world MEC deployments is often proprietary and therefore hard to obtain. However, plausible figures per operator include 2-3 core DCs, hundreds of COs, and tens of thousands of base stations country-wide [45]. Current literature corroborates these numbers. For example, Basta et al. [7] assumed 3 and 4 P-GWs (and thus core DC locations) for Germany and the US, respectively, and Peterson et al. [30] report that, as of 2016, AT&T was operating 4700 COs. Base station approximate locations and densities can be obtained from crowdsourced LTE coverage maps [3, 4, 17]. The resources available in edge DCs will also vary. For example, in the techno-economic analysis of [5, Section 2.3.3.2], a small edge DC was assumed to have a capacity of 576 CPUs, 1512 GB RAM, ~40 TB storage, and ~1000 Gbps network capacity, adding a 5 ms latency.

**Scenario 4 – Vehicular Networks** Services for connected vehicles, including safety, infotainment, and advanced driving assistance, are key drivers for the automotive industry, and can particularly benefit from edge computing. A typical architecture for vehicle-to-everything (V2X) services includes roadside units (RSU) communicating with vehicles’ onboard units using wireless technology and connected with an optical or wireless (e.g., cellular) backhaul to aggregation points, centralized DCs and the Internet. RSUs are similar to the urban sensing nodes we described in Scenario 1, and are candidates for co-location with edge compute hosts for low-latency vehicular services. RSU deployment follows service requirements (coverage, message dissemination latency, reliability). The InterCor project [2] recommends [14] that an RSU should be present at every highway entrance and at least every 6 km. The 5G Automotive Association (5GAA) estimates [6] that the inter-RSU distance will range from 300 m to 1 km, depending on the road type. Due to physical constraints (e.g., an RSU may be mounted at a traffic light in urban environments), the capacity of an edge host deployed at an RSU may be limited, down to that of an SBC or small form-factor node. However, in the Smart Highway testbed (Antwerp, Belgium) [25], some RSUs (less than 1 km apart) are equipped with server-class edge compute nodes, connected connected to the Cloud via fiber links.

### 3.2 Edge Compute Continuum

A challenging aspect of edge computing are the extremes of the compute continuum [39]. For generating nodes in synthetic topologies, we consider the following computers and architectures elicited from the scenarios.

**Edge data centers** that use server computers and are placed at the edge of the network, often termed cloudlets [38], are considered candidate infrastructure for many edge computing scenarios. Typically these cloudlets are considered to be generic VM hosts. More recently, it is assumed that they will carry GPUs for video processing at the edge [40].

**Single Board Computers (SBCs)** such as ARM-based Raspberry Pis are often associated with edge computing, as they can, e.g., function as edge gateways in IoT scenarios to read sensor data, perform data pre-processing or message relaying. Clusters of commodity SBCs have a variety of applications [23]. They are also being used as a platform for hardware AI accelerators such as Google’s Edge TPU [10].

**Small form-factor computers and compute clusters**, such as Intel’s Next Unit of Computing (NUC) platform with built-in CPUs, are used for edge computing scenarios due to their size, low energy footprint, and good performance. For example, the Ubuntu Orange Box is a portable compute cluster consisting of ten NUCs with Intel i5-3428U CPUs [47].

**Embedded AI hardware**, such as NVIDIA’s Jetson TX2, a small computing device with GPU support, are also becoming more relevant for edge computing [29]. These devices enable the acceleration of AI applications at the edge that require computer vision or deep learning workloads.

**Mobile devices** such as smartphones, wearables like AR headsets, smart watches, or health monitoring devices are another category of edge devices [39]. Drones, small factor robots, and vehicle on-board compute elements could also be considered as part of the mobile device spectrum. Typically, mobile devices are equipped with various sensors, whose data are either processed locally or forwarded through a mobile network further up the edge-to-cloud hierarchy.

## 4 A Framework for Synthesizing Edge Computing Infrastructure

Our goal is to make it easy to generate infrastructure configurations and network topologies that can be used as input for simulations for evaluating edge computing systems. *Ether*, the tool we have developed, provides out-of-the box parameterized versions of the scenarios we have defined. It also provides low-level primitives to create high-level building blocks that can be synthesized to generate custom cluster configurations. Figure 1 shows an example of how framework components form a topology for the IIoT scenario.

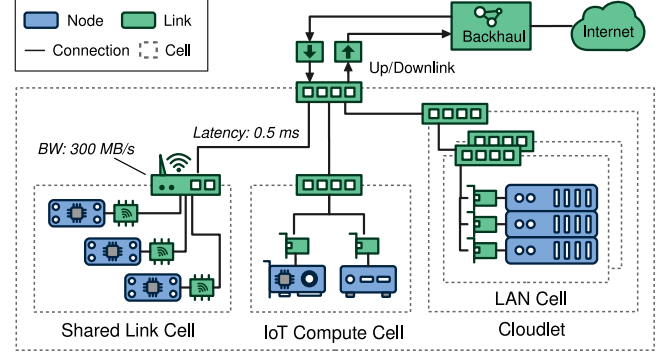


Figure 1: Example of how concepts, primitives, and cell archetypes create a topology for the IIoT scenario

### 4.1 Conceptual Model

The conceptual model of our system is simple, aims to be useful for network or systems simulations, and integrates well with other tools. It comprises the following components.

**Node:** A node is a compute or storage device, and represents an instance of one of the devices listed in Section 3.2. It is characterized by its architecture, generic system capacities (RAM, CPU, storage), and capabilities (presence of GPUs, or other hardware accelerators).

**Link:** A link is anything that facilitates connections between nodes in the network, and can represent network components such as a network card attached to a node, a WiFi access point, or a mobile network base station where bandwidth is allocated across connected nodes. A link has an associated data rate capacity used as parameter for network simulators.

**Topology:** A topology organizes nodes and links as vertices into a graph structure. Edges between nodes and links represent connections, and can hold network-QoS attributes such as latency or jitter. To connect cells across regions, our tool provides a static backhaul graph built from publicly available data on Internet latencies [1].

**Cell:** A cell is a composition of nodes, links or other cells, i.e., an annotated community within the topology. It can represent an edge network that has an up- and a downlink connecting the cell to a backhaul. Cells can be composed of other cells, and connected through links (e.g., switches or routers).

### 4.2 Building Blocks

Most network topology generators provide basic network infrastructure as components. A core contribution of our tool are parameterized cell archetypes as high-level building blocks, that are drawn from our reference scenarios. For example, a cell could be: a RSU node, with two SBCs for sensing and communication, a small mobile base station, and an external connection; or several cloudlets with an average of  $n$  nodes connected through a backhaul.

### 4.2.1 Primitives

Our framework provides the following primitives to compose more complex cells and topologies.

**Host:** The simplest cell is a network host, i.e., a node connected to a network via a link (e.g, the host’s network card). We provide all devices outlined in Section 3.2 as primitives, which can be composed together to form more complex cells. The node’s compute capacities and capabilities are configurable or can be synthesized based on statistical distributions.

**LAN cell:** A LAN cell is a common and simple way of connecting hosts via Point-to-Point connections. The synthesizer creates a new helper node that connects all nodes of the cell, analogous to a switch. It also creates up/down links to connect the cells to the Internet or other cells.

**Shared link cell:** A shared link cell connects specified hosts to a common shared link, whose bandwidth is shared across hosts. This can be used to model a WLAN, a mobile network base station, or VMs sharing the VM-host’s network.

**Geo-spatial cell:** This cell puts nodes and cells into a geo-spatial context, which is useful when evaluating geographically distributed edge systems. For example, RSUs may be uniformly distributed along a road at a fixed distance, whereas USNs and mobile base stations are distributed according to a lognormal distribution across city districts (see Figure 2). These parameters are relevant for, e.g., capacity planning.

**Backhaul connection:** Edge networks are typically connected to the Internet through a backhaul network. Common variants are: (i) direct fiber uplink to an internet exchange with symmetric up/down bandwidth. (ii) business ISP with asymmetric up/down bandwidth in the order of 10-200 MBit/s up, and 100-1000 MBit/s down; (iii) mobile network uplink (e.g., via LTE modem).

### 4.2.2 Cell Archetypes

The primitives of our tool allow custom composition of complex cells. Out-of-the-box, our tool provides pre-made parametric cells that are drawn from the scenarios:

**Cloudlets:** A basic building block for modelling small edge data centers, composed of several racks that can vary in density. Each rack is itself a cell composed of several servers that connect to a switch and may have hardware accelerators such as GPUs. While these may be data-center concepts in terms of terminology, we can use the same concept to build portable multi-purpose cluster-based cloudlets [33], or systems like the Ubuntu Orange Box [47].

**IoT compute cells:** Allows generating small IoT compute boxes, such as urban sensing nodes (USNs) like the AoT or Huawei PoleStar; or RSUs. This building block is similar to a small cloudlet, but much more heterogeneous. Typically they have an IoT gateway connecting sensors or cameras to a processing device with potentially some hardware accelerators, maybe a small storage device, and a communications node connecting the box to a backhaul.

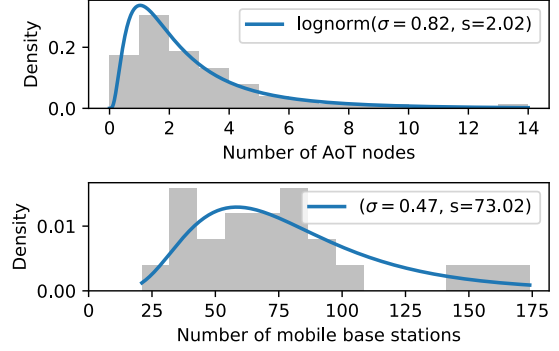


Figure 2: Example geographic cell density functions. Top (Scenario 1): Density of AoT nodes per neighborhood in Chicago. Bottom (Scenario 3): Density of mobile base stations per district in Vienna, Austria [17].

### 4.2.3 Parameterized Cell Synthesis

Cells can either be defined statically, or synthesized dynamically from parameters to create randomized cells. Parameterized synthesis is what allows us to generate graphs that are similar across scenarios, but different enough to test different possible variations of a system. Such parameters include:

**Size:** The size  $n$  specifies the number of nodes or cells in the final cell’s topology. In other words the cell is made up of  $n$  other cells that are created by a specified procedure.

**Density:** If a cell is composed of other synthesized cells, we can provide a density function from which  $n$  is sampled for each new cell being created. For example, Figure 2 shows parameters for the density of cells in the Chicago city AoT deployment and mobile base stations in Vienna.

**Entropy:** borrows from information entropy to describe the heterogeneity of cells, i.e., the number and density of different compute device types, and how much nodes differ in compute capacity. A low entropy value means that the cell is fairly homogeneous (e.g., 0 for a server rack with one type of server), whereas high entropy describes heterogeneous cells. This parameter is useful for evaluating, e.g., resource allocation mechanisms under different levels of cluster heterogeneity.

**Data distribution:** Many edge computing systems evaluations deal with data aspects such as distributed storage, caching, etc. An important parameter for these evaluations is the distribution of data across storage nodes, and cells. Like cell density, we can define distribution functions to randomly vary the availability of storage nodes across edge networks.

### 4.2.4 Connecting Cells to the Internet

Many systems evaluate network latencies incurring from cross-regional Internet traffic. Compared to application and use-case specific networks, the core Internet has a fairly static layout, with well-studied link latencies [1], which we can leverage. Our tool provides a static fixture of the core Internet,



to which application-specific topologies can be linked. For example, in our IIoT scenario, we could link the premises' up- and downlinks to the respective Internet region in which the premises are located. That way, user-generated topologies receive coarse-grained latency values out-of-the-box.

## 5 Demonstration

We demonstrate how our tool *ether* can generate topologies for the reference scenarios from Section 3, and how topologies can be used to analyze system behavior such as network flows.

### 5.1 Generating Configurations

Our tool provides a pre-made parameterized version of **Scenario 1** (urban sensing), but we give an example of how a topology can be generated for it using our tool. We extend the scenario and model it as follows. Each AoT node is an IoT Compute Cell with two SBCs, and connected to a mobile base station in the neighborhood (i.e., they form a shared link cell). For the number of AoT nodes in each of the 77 neighborhoods we use the geographic cell density from Figure 2. We assume that each neighborhood has a compute box attached to the mobile base station with an Intel NUC as storage node, and two embedded GPU devices per AoT node camera for, e.g., video processing tasks. The cells are connected via mobile network to the Internet (in terms of Internet topology, as explained in Section 4.2.4, we connect them to the Chicago Internet Exchange). Furthermore, the city provides a fiber connected cloudlet with two racks and five servers per rack. The Python code to generate this topology is given in Listing 1. We make use of Python lambdas for parameterized synthesis, and API methods to materialize cells into a topology.

```
aot_node = IoTComputeBox(nodes=[nodes.rpi3, nodes.rpi3])
neighborhood = lambda size: SharedLinkCell(
    nodes=[
        [aot_node] * size,
        IoTComputeBox([nodes.nuc]+([nodes.tx2] * size*2))
    ],
    shared_bandwidth=800, # selected arbitrarily
    backhaul=MobileConnection('internet_chix')
)
city = GeoCell(
    77, nodes=[neighborhood], density=lognorm(0.82, 2.))
cloudlet = Cloudlet(
    5, 2, backhaul=FiberToExchange('internet_chix'))

topology = Topology()
topology.add(city)
topology.add(cloudlet)
```

Listing 1: Code to implement the urban sensing scenario

### 5.2 Using Topologies in Simulations

As part of the research presented in [35], we have built a tool to simulate serverless and containerized platforms [37], which we use to evaluate different placement strategies in different infrastructure scenarios. We use *ether* to generate several topologies as input for our simulation. The simulator implements a high-level network model on top of topologies based

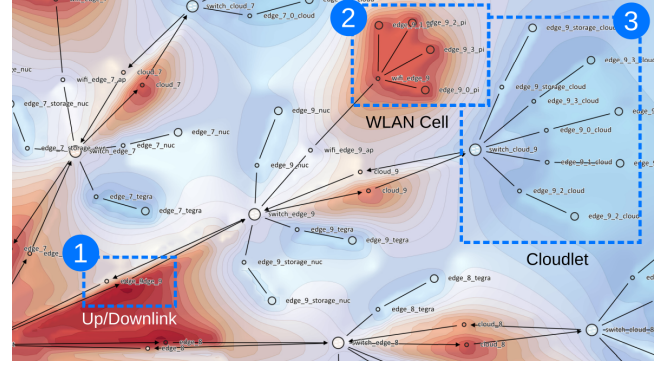


Figure 3: Visualization of a generated IIoT scenario topology and simulation data from [35] using a *topographic attribute map* [31]. The elevation field shows link capacity utilization caused by data transfer between serverless functions [35].

around *flows*, which represent data transfers between nodes through several links. We implemented a simple shortest-path routing through the topology and fair allocation of link bandwidth across flows.

The following example is an instance of **Scenario 2** (Industrial IoT) based on a prototype presented in [13]. We assume ten factory locations, each having 4 SBCs as IoT gateways, and a compute box with 1 Intel NUC and 1 Jetson TX2 board; as well as an on-prem cloudlet with 5 servers, as represented in Figure 1. The SBCs are connected via a shared 300MBit/s WiFi link to an AP that has a 10 GBit/s link to the edge resources, and a 1Gbit/s link to the on-prem cloud. Premises are connected via 250/500 MBit/s Internet up/downlinks.

Figure 3 shows the data traffic of a particular placement in the above configuration. (1) Shows the up/downlink connecting the factory premises to the backhaul. (2) Shows the shared link cell that contains the four SBC devices connected to a shared WiFi link. The elevation field shows that the 300 MBit/s bandwidth is exhausted. (3) Shows the cloudlet which, given the 10 GBit/s internal LAN, has the lowest utilization.

## 6 Conclusion

Evaluating edge computing systems, such as compute platforms or resource allocation algorithms, in a generalized way is challenging. Although existing tools facilitate simulation or emulation of such systems, it is difficult to describe an underlying edge infrastructure and vary its parameters. We presented edge infrastructure scenarios that encompass many different edge system characteristics. We abstracted these characteristics into a framework and a tool, and showed how it can generate different infrastructure configurations for these scenarios, and how the building blocks we provide can help edge systems researchers evaluate systems under different scenarios and conditions. For future work we are investigating how to model the dynamics of topologies over time, e.g., for mobile scenarios where nodes move through the network.

## Discussion Topics

In keeping with the workshop format, in this section we discuss a) what kind of feedback we are looking to receive b) the controversial points of the paper c) the type of discussion this paper is likely to generate in a workshop format d) the open issues the paper does not address, and e) under what circumstances the whole idea might fall apart

- a) We do not claim to have found an exhaustive list of edge computing systems scenarios, and are therefore very interested to learn what other scenarios and related infrastructure the edge systems community works with, and what features would be useful to them. We hope to learn whether the scenarios we have presented, as well as the selected parameter space and values, are appropriate and plausible, and what other parameters are important for evaluations. Overall we are trying to gauge how useful the presented idea could be for the community.
- b) The idea of federating distributed resources into a compute fabric for edge computing systems is not a widely accepted model, and may only work for a certain type of application. Furthermore, although we have done our best to corroborate Section 3 with sources, the scenarios and the concrete numbers we have presented require a certain degree of speculation, which is, inevitably, the dilemma that we often face when developing and evaluating edge computing systems, given the nascence of the field.
- c) We hope to spark a discussion about i) the available methodologies for evaluating edge computing systems in general, and what we as a community can do to improve them to allow more generalization of results; and ii) which edge computing infrastructure scenarios people see as most relevant for future research and development.
- d) Our paper does not address the modeling of topology dynamics over time. Edge computing scenarios often have moving parts, especially in mobile systems, where nodes and potential resources move through the network. Modeling and describing those dynamics, e.g., the trajectories of cognitive assistance device users through a city, could, however, be very useful. Furthermore, we do not cover application or system-specific parameters such as workload characteristics or request arrival patterns.
- e) It is possible that the programming model we propose is too tailored to the use cases we are working with, and that it does not generalize to other people's research the way we would like it to. We have not fully explored the integration with other common simulation or emulation tools, and it is unclear how easy or difficult it will be to facilitate interoperability.

## Acknowledgements

We thank Reinhold Preiner for creating the beautiful visualization of our topology and simulation data. This work is supported by the Austrian infrastructure program (HRSM 2016) as part of the CPS/IoT Ecosystem project.

## References

- [1] AWS inter-region latency monitoring. <https://www.cloudping.co/>.
- [2] InterCor: Interoperable corridors deploying cooperative intelligent transport systems, connecting europe facility (CEF) programme (2014-2020). <https://intercor-project.eu/>.
- [3] OpenCellid. <https://opencellid.org>.
- [4] Opensignal coverage maps. <https://www.opensignal.com/networks>.
- [5] 5G-TRANSFORMER project. 5G-TRANSFORMER final system design and techno-economic analysis. Public Deliverable D1.4, November 2019.
- [6] 5GAA. C-ITS vehicle to infrastructure services: How C-V2X technology completely changes the cost equation for road operators. White paper, 5G Automotive Association, January 2019.
- [7] Arsany Basta, Andreas Blenk, Klaus Hoffmann, Hans Jochen Morper, Marco Hoffmann, and Wolfgang Kellerer. Towards a cost optimal design for a 5G mobile core network based on SDN and NFV. *IEEE Trans. Network and Service Management*, 14(4):1061–1075, 2017.
- [8] Pete Beckman. Artificial intelligence at the edge: How machine learning at the edge is creating a computing continuum. Presentation. [https://www.youtube.com/watch?v=N\\_k8Uh8B10E](https://www.youtube.com/watch?v=N_k8Uh8B10E), 2018. Research Computing Centre.
- [9] Kenneth L Calvert, Matthew B Doar, and Ellen W Zegura. Modeling internet topology. *IEEE Communications Magazine*, 35(6):160–163, 1997.
- [10] Stephen Cass. Taking AI to the edge: Google's TPU now comes in a maker-friendly package. *IEEE Spectrum*, 56(5):16–17, 2019.
- [11] Charles E Catlett, Peter H Beckman, Rajesh Sankaran, and Kate Kusiak Galvin. Array of things: A scientific research instrument in the public way: Platform design and early lessons learned. In *Proceedings of the 2nd*

*International Workshop on Science of Smart City Operations and Platforms Engineering*, pages 26–33. ACM, 2017.

- [12] Angelo Cenedese, Andrea Zanella, Lorenzo Vangelista, and Michele Zorzi. Padova smart city: An urban internet of things experimentation. In *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, pages 1–6. IEEE, 2014.
- [13] Baotong Chen, Jiafu Wan, Antonio Celesti, Di Li, Haider Abbas, and Qin Zhang. Edge computing in IoT-based manufacturing. *IEEE Communications Magazine*, 56(9):103–109, 2018.
- [14] Yvonne Dierikx-Platschorre, Jaap Vreeswijk, Anton Wjibenga, Patrick Hofman, and Ruben van Ardenne. Guideline placement C-ITS roadside units. In *Proc. 12th ITS European Congress*, 2017.
- [15] Paul Erdős and Alfréd Rényi. On random graphs I. *Publicationes Mathematicae Debrecen*, 6(290-297):18, 1959.
- [16] ETSI GS MEC 003. *Multi-Access Edge Computing (MEC); Framework and Reference Architecture*, January 2019. V2.1.1.
- [17] Federal Ministry for Climate Action, Environment, Energy, Mobility, Innovation and Technology. Senderkataster Austria. <https://www.senderkataster.at/>.
- [18] Damián Fernández-Cerero, Alejandro Fernández-Montes, F Javier Ortega, Agnieszka Jakóbiak, and Adrian Widlak. Sphere: Simulator of edge infrastructures for the optimization of performance and resources energy consumption. *Simulation Modelling Practice and Theory*, page 101966, 2019.
- [19] Harshit Gupta, Amir Vahid Dastjerdi, Soumya K Ghosh, and Rajkumar Buyya. iFogSim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments. *Software: Practice and Experience*, 47(9):1275–1296, 2017.
- [20] Kiryong Ha, Zhuo Chen, Wenlu Hu, Wolfgang Richter, Padmanabhan Pillai, and Mahadev Satyanarayanan. Towards wearable cognitive assistance. In *Proceedings of the 12th annual international conference on Mobile systems, applications, and services*, pages 68–81, 2014.
- [21] Ilija Hadžić, Yoshihisa Abe, and Hans C Woithe. Edge computing in the ePC: A reality check. In *Proceedings of the Second ACM/IEEE Symposium on Edge Computing, SEC '17*, pages 1–10, 2017.
- [22] Kai Hwang, Xiaoying Bai, Yue Shi, Muyang Li, Wenguang Chen, and Yongwei Wu. Cloud performance modeling with benchmark evaluation of elastic scaling strategies. *IEEE Transactions on parallel and distributed systems*, 27(1):130–143, 2015.
- [23] Steven J Johnston, Philip J Basford, Colin S Perkins, Herry Herry, Fung Po Tso, Dimitrios Pazaros, Robert D Mullins, Eiko Yoneki, Simon J Cox, and Jeremy Singer. Commodity single board computer clusters and their applications. *Future Generation Computer Systems*, 89:201–212, 2018.
- [24] Simone Mangiante, Guenter Klas, Amit Navon, Zhuang GuanHua, Ju Ran, and Marco Dias Silva. Vr is on the edge: How to deliver 360 videos in mobile networks. In *Proceedings of the Workshop on Virtual Reality and Augmented Reality Network*, pages 30–35, 2017.
- [25] Johann Marquez-Barja, Bart Lannoo, Dries Naudts, Bart Braem, Carlos Donato, Vasilis Maglogiannis, Siegfried Mercelis, Rafael Berkvens, Peter Hellinckx, Maarten Weyn, Ingrid Moerman, and Steven Latre. Smart highway: ITS-G5 and C-V2X based testbed for vehicular communications in real environments enhanced by edge/cloud technologies. In *Proc. 28th European Conference on Networks and Communications (EuCNC)*, 2019.
- [26] Ruben Mayer, Leon Graser, Harshit Gupta, Enrique Saurez, and Umakishore Ramachandran. Emufog: Extensible and scalable emulation of large-scale fog computing infrastructures. In *2017 IEEE Fog World Congress (FWC)*, pages 1–6. IEEE, 2017.
- [27] Alberto Medina, Anukool Lakhina, Ibrahim Matta, and John Byers. BRITE: An approach to universal topology generation. In *MASCOTS 2001, Proceedings Ninth International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pages 346–353. IEEE, 2001.
- [28] Flávio Nunes. Aveiro, portugal: Making a digital city. *Journal of Urban Technology*, 12(1):49–70, 2005.
- [29] NVIDIA. NVIDIA Jetson - the AI platform for autonomous machines. <https://developer.nvidia.com/embedded/develop/hardware>.
- [30] Larry L. Peterson, Ali Al-Shabibi, Tom Anshutz, Scott Baker, Andy C. Bavier, Saurav Das, Jonathan Hart, Guru Palukar, and William Snow. Central office re-architected as a data center. *IEEE Communications Magazine*, 54(10):96–101, 2016.

- [31] Reinhold Preiner, Johanna Schmidt, Katharina Krösl, Tobias Schreck, and Gabriel Mistelbauer. Augmenting node-link diagrams with topographic attribute maps. *Computer Graphics Forum*, May 2020.
- [32] Thomas Rausch. ether: edge topology synthesizer. <https://github.com/edgerun/ether>.
- [33] Thomas Rausch, Cosmin Avasalcai, and Schahram Dustdar. Portable energy-aware cluster-based edge computers. In *2018 IEEE/ACM Symposium on Edge Computing (SEC)*, pages 260–272, 2018.
- [34] Thomas Rausch and Schahram Dustdar. Edge intelligence: The convergence of humans, things, and AI. In *2019 IEEE International Conference on Cloud Engineering (IC2E'19)*, Jun 2019.
- [35] Thomas Rausch, Waldemar Hummer, Vinod Muthusamy, Alexander Rashed, and Schahram Dustdar. Towards a serverless platform for edge AI. In *2nd USENIX Workshop on Hot Topics in Edge Computing (HotEdge 19)*, 2019.
- [36] Thomas Rausch, Stefan Nastic, and Schahram Dustdar. EMMA: Distributed QoS-aware MQTT middleware for edge computing applications. In *2018 IEEE International Conference on Cloud Engineering (IC2E)*, pages 191–197. IEEE, 2018.
- [37] Thomas Rausch and Alexander Rashed. FaaS-Sim: serverless scheduler simulation. <https://git.dsg.tuwien.ac.at/tau/faas-sim>.
- [38] Mahadev Satyanarayanan, Victor Bahl, Ramón Caceres, and Nigel Davies. The case for VM-based cloudlets in mobile computing. *IEEE pervasive Computing*, 2009.
- [39] Mahadev Satyanarayanan, Wei Gao, and Brandon Lucia. The computing landscape of the 21st century. In *Proceedings of the 20th International Workshop on Mobile Computing Systems and Applications*, pages 45–50, 2019.
- [40] Pieter Simoens, Yu Xiao, Padmanabhan Pillai, Zhuo Chen, Kiryong Ha, and Mahadev Satyanarayanan. Scalable crowd-sourcing of video from mobile devices. In *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services, MobiSys '13*, page 139–152, New York, NY, USA, 2013. Association for Computing Machinery.
- [41] Olena Skarlat, Matteo Nardelli, Stefan Schulte, and Schahram Dustdar. Towards QoS-aware fog service placement. In *2017 IEEE 1st international conference on Fog and Edge Computing (ICFEC)*, pages 89–96. IEEE, 2017.
- [42] Cagatay Sonmez, Atay Ozgovde, and Cem Ersoy. Edge-cloudsim: An environment for performance evaluation of edge computing systems. In *2017 Second International Conference on Fog and Mobile Edge Computing (FMEC)*, pages 39–44. IEEE, 2017.
- [43] Pablo Sotres, Juan Ramón Santana, Luis Sánchez, Jorge Lanza, and Luis Muñoz. Practical lessons from the deployment and management of a smart city internet-of-things infrastructure: The smartsantander testbed case. *IEEE Access*, 5:14309–14322, 2017.
- [44] Hongsuda Tangmunarunkit, Ramesh Govindan, Sugih Jamin, Scott Shenker, and Walter Willinger. Network topology generators: Degree-based vs. structural. *ACM SIGCOMM Computer Communication Review*, 32(4):147–159, 2002.
- [45] Telefonica. Telefonica open access and edge computing. White paper, February 2019.
- [46] Liang Tong, Yong Li, and Wei Gao. A hierarchical edge cloud architecture for mobile computing. In *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9. IEEE, 2016.
- [47] Steven J. Vaughan-Nichols. Canonical's cloud-in-a-box: The ubuntu orange box. ZDNet. <https://zdnet.com/article/canonicals-cloud-in-a-box-the-ubuntu-orange-box>.
- [48] Abhishek Verma, Luis Pedrosa, Madhukar Korupolu, David Oppenheimer, Eric Tune, and John Wilkes. Large-scale cluster management at google with borg. In *Proceedings of the Tenth European Conference on Computer Systems*, page 18. ACM, 2015.
- [49] B. M. Waxman. Routing of multipoint connections. *IEEE Journal on Selected Areas in Communications*, 6(9):1617–1622, Dec 1988.
- [50] Xiaolong Xu, Daoming Li, Zhonghui Dai, Shancang Li, and Xuening Chen. A heuristic offloading method for deep learning edge services in 5g networks. *IEEE Access*, 7:67734–67744, 2019.
- [51] Jihong Yan, Yue Meng, Lei Lu, and Lin Li. Industrial big data in an industry 4.0 environment: Challenges, schemes, and applications for predictive maintenance. *IEEE Access*, 5:23484–23491, 2017.