# Conventional and Machine Learning Approaches for Network and Service Coordination

Stefan Schneider[§]
Paderborn University, Germany
stefanbschneider@outlook.com

***Abstract*—Network and service coordination is important to provide modern services consisting of multiple interconnected components, e.g., in 5G, network function virtualization (NFV), or cloud and edge computing. In this paper, I outline my dissertation research, which proposes several approaches to automate such network and service coordination. All approaches dynamically react to the current demand and optimize coordination for high service quality and low costs. The approaches range from centralized to distributed methods and from conventional heuristic algorithms and mixed-integer linear programs to machine learning approaches using supervised and reinforcement learning. I briefly discuss their main ideas and advantages over other state-of-the-art approaches and compare strengths and weaknesses.**

## I. Introduction

In various practical scenarios, services consist of multiple chained components, where each component is implemented in software and provides its own functionality. For example, components could be virtual network functions (VNFs) in network function virtualization (NFV) [2] and 5G or 6G [3], microservices in a service mesh for cloud and edge computing [4], or machine learning functions in a pipeline [5]. To deploy these services according to the current demand, each component can be instantiated multiple times across different nodes in the network. Nodes are distributed at different locations, where each node may represent a large data center, a small edge server, or a node without compute capacity. These nodes are interconnected by links with varying data rate limitations.

As user demand for the different services arrives at the network's ingress nodes, *network and service coordination* ensures that the requested services are deployed and provided with high quality and low costs. Particularly, I distinguish four main coordination aspects as illustrated in Fig. 1: 1) Service scaling, which determines the number of required instances per component. 2) Service placement, where suitable nodes are selected for deploying these instances. 3) Flow scheduling, where incoming flows are assigned to the placed instances. 4) Routing, which determines the flows' path between users and the assigned instances.

These four coordination aspects are inter-dependent and should therefore be optimized jointly [6]. As they depend on a multitude of parameters, it is challenging to ensure that enough resources are allocated for good service quality but also no resources and costs are wasted. With demand changing over time, coordination has to happen dynamically online.

This paper presents and discusses the work of my dissertation [1] (completed in January 2022), which proposes approaches for automated and dynamic network and service coordination to address the aforementioned challenges. I group these approaches into either conventional approaches like heuristics and mixed-integer linear programs (MILPs) or into machine learning approaches that learn from available data. For both kinds of approaches, I discuss state of the art and related work in Sec. II. Sec. III defines the problem statement of network and service coordination. Sec. IV and V present the various conventional and machine learning approaches to solve the problem, which are the main contribution of my dissertation. Finally, Sec. VI discusses the potential impact of these contributions and concludes this paper.

## II. Related Work

### A. Related Conventional Approaches

Herrera and Botero [2] as well as Kaur et al. [7] survey state-of-the-art approaches for network and service coordination. The large majority of this existing work proposes conventional coordination approaches that are manually designed by human experts, precisely defining MILP formulations or heuristic algorithms according to their knowledge and understanding of a given scenario or problem. In turn, most of these conventional approaches assume detailed and global knowledge and control of network and services. For example, Moens and De Turck [8] propose a centralized optimization approach for placing VNF instances in a hybrid setting, coexisting with physical network functions. Unlike the approaches proposed in my dissertation, the authors do not consider dynamic scaling of services or traffic routing between component instances. Similarly, other related approaches [9]–[11] focus on placement of instances and/or traffic routing but disregard flexible scaling of services according to the current demand.

In contrast, my proposed coordination approach "BSP" (Sec. IV-A) jointly optimizes service scaling and placement as well as flow scheduling and routing. In addition to BSP, which is a centralized coordination approach, I also propose a hierarchical and two fully distributed coordination approaches (Sec. IV-B and IV-C). The benefit of these approaches over typical centralized approaches is that they do not require global knowledge and control of the entire network and all involved

[§]Dissertation completed in January 2022 at Paderborn University [1].
Current affiliation: Digital Rail for Germany (Deutsche Bahn).
PhD supervisor: Prof. Dr. Holger Karl (holger.karl@hpi.de)

services. Furthermore, they scale better to large networks, making them more realistically applicable.

There is comparably little related work on hierarchical or distributed coordination approaches. In virtual network embedding (VNE), which is closely related to network and service coordination, Samuel et al. [12] and Ghazar et al. [13] propose hierarchical approaches. In contrast to my proposed hierarchical approach (Sec. IV-B), theirs do not aggregate or abstract any information such that computations on high hierarchical levels become very expensive, comparable to centralized approaches. Similarly, multiple authors propose distributed approaches for VNE [14], [15], which split the network into clusters that process incoming flows in parallel. My proposed approaches in Sec. IV-C go one step further and coordinate service scaling and placement as well as flow scheduling and routing in a fully distributed manner – each node making fast and local coordination decisions by itself. In doing so, these approaches are based on information that is locally available and are even more scalable then existing distributed approaches.

## B. Related Machine Learning Approaches

In recent years, a growing amount of research has been focusing on machine learning for network and service coordination [16]. Some authors propose machine learning approaches that support and improve conventional coordination approaches, e.g., by predicting traffic demands [17], VNF instances' processing delays [18], or their resource requirements [19]. Complementing and going beyond this research, I propose a machine learning approach for learning from real-world performance data to precisely allocate required resources (Sec. V-A). Integrating this machine learning approach with a conventional coordination approach results in significantly fewer wasted resources while maintaining high service quality.

Other research considers self-learning coordination approaches using deep reinforcement learning (DRL) [20]. Some existing approaches combine DRL with a conventional heuristic [21], [22], others rely on a priori traffic knowledge as input for their DRL approaches [23]. To the best of my knowledge, my "DeepCoord" approach (Sec. V-B) is the first self-learning DRL approach for network and service coordination that works with realistically available partial and delayed observations. It jointly optimizes scaling, placement, and scheduling without requiring support from a heuristic, making it more versatile and less error-prone than existing approaches. I also propose one of the first distributed coordination approaches using DRL (Sec. V-C), where each node coordinates scaling, placement, scheduling, and routing autonomously. This distributed approach allows fast individual coordination decisions for rapidly incoming flows, scaling to large networks. Finally, I propose a suite of centralized and distributed DRL approaches for multi-cell selection in wireless mobile networks (Sec. V-D). These approaches complement existing DRL approaches for coordinating wireless mobile networks, e.g., for dynamic power control [24] or resource allocation [25].

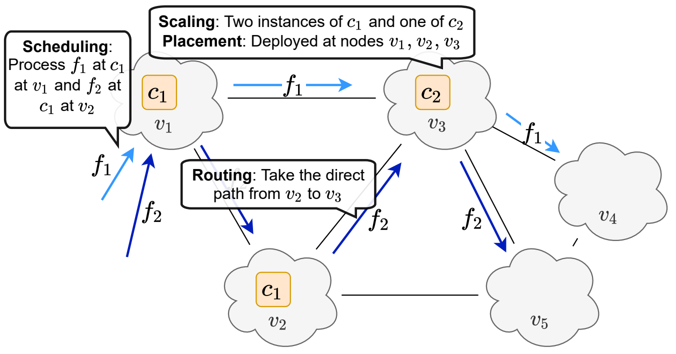

**Scheduling**: Process $f_1$ at $c_1$ at $v_1$ and $f_2$ at $c_1$ at $v_2$
**Scaling**: Two instances of $c_1$ and one of $c_2$
**Placement**: Deployed at nodes $v_1, v_2, v_3$
**Routing**: Take the direct path from $v_2$ to $v_3$
$c_1$
$v_1$
$c_2$
$v_3$
$c_1$
$v_2$
$v_4$
$v_5$
$f_1$
$f_1$
$f_1$
$f_1$
$f_2$
$f_2$
$f_2$

Fig. 1: Four main aspects of network and service coordination.

## III. Problem Statement

This section defines the parameters, decision variables, and optimization objectives of network and service coordination.

### A. Problem Parameters

The problem parameters are given by the network, the services, and the incoming traffic as detailed next. The network contains distributed compute nodes interconnected by links. As such, it can be modeled as graph $G = (V, L)$, where each node $v \in V$ has limited compute capacity $\text{cap}_v$, e.g., CPU cores, and each link $l \in L$ has a limited data rate $\text{cap}_l$ and a propagation delay $d_l$.

In line with the IETF's definition of service function chaining (SFC) [26], I define services as linear chains of interconnected components, e.g., representing VNFs, microservices, or machine learning functions. Each component $c \in C_s$ of service $s \in S$ can be instantiated zero, one, or multiple times and deployed across different nodes in the network, depending on the current demand. Each instance of $c$ has identical functionality and requires compute resources $r_c(\lambda)$ depending on the data rate $\lambda$ it has to process.

Traffic for these services arrives in the form of flows at the network's ingress nodes over time. Each flow $f = (s_f, v_f^{\text{in}}, v_f^{\text{eg}}, \lambda_f, t_f^{\text{in}}, \delta_f) \in F$ is characterized by its requested service $s_f \in S$, its ingress and egress nodes $v_f^{\text{in}}, v_f^{\text{eg}} \in V$, its data rate $\lambda_f$, its time of arrival $t_f^{\text{in}}$, and its duration $\delta_f$. For successfully processing a flow, it has to be steered from its ingress node through instances of all requested service components to its egress node.

### B. Decision Variables

As mentioned in Sec. I and illustrated in Fig. 1, I consider four main aspects of network and service coordination: 1) Scaling, 2) placement, 3) scheduling, and 4) routing.

Service scaling and placement determines how many instances to deploy of each component $c$ and at which nodes to place them. Binary decision variable $x_{c,v}(t) \in \{0, 1\}$ indicates whether or not an instance of $c$ is placed at node $v$ at time $t$. In the example of Fig. 1, $x_{c_1,v_1} = x_{c_1,v_2} = x_{c_2,v_3} = 1$. I focus on *inter*-node coordination and assume that coordination *within* a node is controlled transparently by the node's operating system or systems like Kubernetes [27].

Flow scheduling is determined by decision variable $y_{f,c}(t) \in V \cup \{\varnothing\}$, which indicates at which node $v$ (if any) to process flow $f$ that requests component $c$ at time $t$. In Fig. 1, flow $f_1$ requests $c_1$ and is scheduled to the instance of $c_1$ at node $v_1$, i.e., $y_{f_1,c_1}(t^{\text{in}}_{f_1}) = v_1$. Finally, routing determines the path between different instances and ingress and egress. Here, decision variable $z_{f,v}(t) \in V \cup \{\varnothing\}$ indicates where to route flow $f$ next that is currently at node $v$. In Fig. 1, flow $f_2$ is routed from $v_2$ to $v_3$ directly, i.e., $z_{f_2,v_2}(t) = v_3$.

### C. Optimization Objectives

Many optimization objectives are conceivable for network and service coordination. As main objective, I focus on providing the requested services to as many users as possible by steering and processing flows successfully without exceeding capacities. As additional objective, I consider minimizing flows' end-to-end delay, i.e., from entering the network to traversing all service component instances and finally departing the network again. Short end-to-end delay is important for good Quality of Service (QoS) but may require more instances and resources, thus potentially conflicting with the main objective. In some of my work, I also consider minimizing the number of deployed instances or the number of active compute nodes to reduce costs.

## IV. Conventional Approaches

Conventional coordination approaches, e.g., heuristics or MILPs, are designed by experts and build on detailed models tailored to specific scenarios. They are suitable for scenarios that are well understood and can be modeled and observed accurately. In this section, I briefly present three types of conventional coordination approaches: A centralized heuristic (Sec. IV-A), a hierarchical approach (Sec. IV-B), and two fully distributed approaches (Sec. IV-C). For each approach, I reference the corresponding paper containing details and information about co-authors.

### A. Centralized Coordination [28]

I propose a heuristic algorithm (called BSP) for joint optimization of all four coordination aspects, even supporting services with complex bidirectional structures [28]. The BSP algorithm starts with an initialization phase, which precomputes the shortest paths between all pairs of nodes, taking link delay and capacity into account. Afterwards, the algorithm's core part, the embedding procedure, is executed repeatedly following a destroy-and-repair approach using tabu search. The embedding procedure sequentially handles incoming flows, scheduling them to existing instances or placing new instances, depending on available resources and taking end-to-end delay into account. Through the repeated execution with tabu search, the initial solution is improved iteratively, overcoming potential local optima.

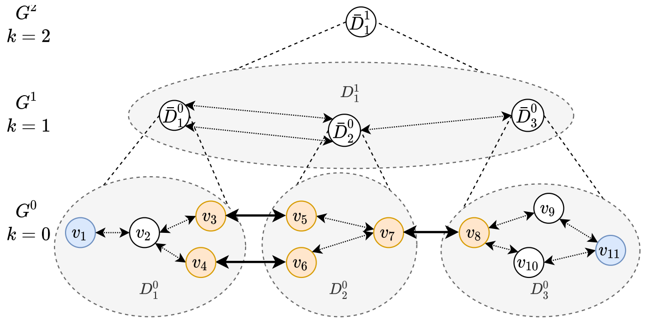


Fig. 2: Hierarchical coordination aggregating lower-level details into domains controlled by higher-level coordinators. (Figure from [29]; ©2021 IEEE.)

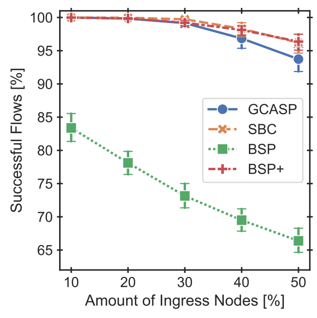
The evaluation (detailed in [28]) on real-world network topologies shows that the BSP heuristic achieves close to optimal results, when compared against an optimal MILP solution. While it requires more instances to process many flows, it leverages these instances to achieve even lower end-to-end delay than the MILP solution. It accurately scales the deployed services to the current demand, handling increasing demand by placing more instances and allocating more resources and then removing them again when demand decreases. In doing so, the BSP heuristic is *magnitudes* faster than the comparable MILP approach (e.g., $6.7\,\mathrm{s}$ compared to $137.1\,\mathrm{h}$).

### B. Hierarchical Coordination [29]

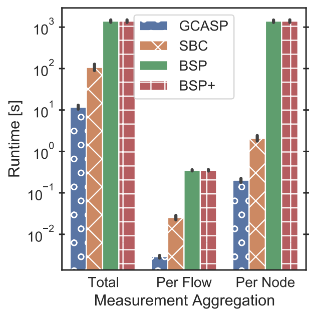
In large networks, detailed up-to-date global knowledge may not be available and centralized decisions may be too inefficient, even with a heuristic. Therefore, I propose a hierarchical coordination approach, which works in two phases [29]: In the bottom-up phase, it aggregates information from lower hierarchical levels, hiding unnecessary details from higher-level coordinators to reduce complexity as illustrated in Fig. 2. In the following top-down phase, higher-level coordinators make coarse-grained coordination decisions that are then refined at lower levels. I implement the hierarchical approach as an MILP, but the general framework can also be applied to heuristics or other optimization approaches. Compared to an equivalent centralized approach, this hierarchical approach finds solutions with similar quality but is significantly faster.

### C. Fully Distributed Coordination [30]

Going one step further, I propose two simple and fast heuristics for fully distributed coordination [30]. The algorithms are executed independently in parallel at each node in the network and rely only on local observations and control at each node. This allows fast-paced online coordination in practical large-scale networks with rapidly arriving flows. One algorithm is greedy (called GCASP) and designed to be simple, effective, fast, and frugal, sending flows along the shortest path but avoiding congestion. The other algorithm (called SBC) can take further information from neighboring nodes into account to compute node scores and achieve higher solution quality.



(a) Solution quality



(b) Runtime (log. scale)

Fig. 3: The fully distributed heuristics (GCASP, SBC) reach comparable solution quality but are much faster than the centralized BSP and BSP+. (Figure from [30]; ©2020 IEEE.)

According to the evaluation [30], the fully distributed heuristics achieve similar solution quality to the centralized BSP+ approach (Fig. 3a). BSP+ is an adjusted version of BSP (Sec. IV-A) that, unlike BSP, can properly handle rapidly arriving and partially overlapping flows. However, the fully distributed heuristics make much simpler and faster coordination decisions in parallel at each node, leading to significantly faster runtimes (Fig. 3b).

## V. Machine Learning Approaches

Machine learning coordination approaches rely less on expert knowledge but learn coordination from available data or their own experience without human intervention. Again, this section briefly presents my proposed coordination approaches using machine learning. Details are in the referenced papers.

### A. *Machine Learning for Dynamic Resource Allocation [31]*

Most coordination approaches either allocate fixed resources per instance or rely on simple, predefined functions to estimate and allocate the required resources $r_c(\lambda)$ of an instance of service component $c$ with current load $\lambda$. As it is hard to accurately model real-world components and their resource requirements manually, I propose an approach using supervised learning that automatically derives these models from available real-world benchmarking data [31].

In the evaluation [31], I compare six different machine learning approaches to learn from real-world VNF benchmarking data [32] and predict their required resources under varying load. I show that integrating the trained machine learning models into the conventional BSP algorithm (Sec. IV-A) requires low overhead but can significantly improve coordination performance, effectively avoiding over- and under-allocation of resources.

### B. *Self-Learning Centralized Coordination [33], [34]*

As an alternative to conventional approaches, I propose a novel and completely autonomous self-learning coordination approach using model-free DRL called DeepCoord [33], [34]

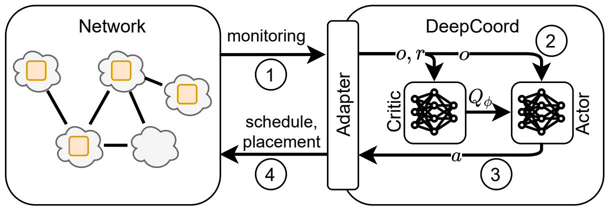


Fig. 4: DeepCoord iteratively interacts with the network through an adapter. (Figure from [34]; ©2021 IEEE.)

(Best Student Paper at CNSM 2020 and invited talk at IETF 110). The actor-critic approach trains a central DRL agent based on realistically available partial and delayed observations from monitoring. The trained approach then quickly and periodically interacts with the network by obtaining monitoring data and updating service scaling and placement as well as coordination rules for flow scheduling (Fig. 4). The scheduling rules provided by DeepCoord are deployed at all network nodes and applied to incoming flows locally at runtime.

In doing so, DeepCoord consistently outperforms existing approaches, requiring significantly fewer resources to ensure high success rates on real-world network topologies. It self-adapts to varying scenarios without human intervention, is robust to change, generalizes to unseen traffic patterns, learns to optimize multiple objectives, and scales to networks of realistic size while only relying on information that is available in practice. Specifically, I observe that DeepCoord reaches up to 76% more successful flows and more than 2x higher total utility than BSP (Sec. IV-A) when optimizing multiple objectives.

### C. *Self-Learning Distributed Coordination [35]*

Combining ideas from the fully distributed heuristics in Sec. IV-C and DeepCoord (Sec. V-B), I propose a distributed self-learning coordination approach using DRL [35]. After centralized training, it deploys separate DRL agents at each node in the network, coordinating incoming flows individually and in parallel but using self-learning DRL agents rather than hand-written algorithms. As such, the approach combines the benefits of both Sec. IV-C and V-B: It only requires local observations and control, scales independently from the network size (depends only on the network degree), allows fast, fine-grained per-flow control, and self-adapts to new and unseen scenarios without expert knowledge or human intervention. Specifically, the DRL agent at a node decides for each incoming flow whether to process the flow locally (potentially starting a new instance) or to forward it to one of the neighboring nodes, taking available instances, delays, and resource utilization into account (Fig. 5).

Thanks to the fast per-flow control, this distributed DRL approach achieves even higher success rates than the centralized DeepCoord approach, which works with aggregated observations and more coarse-grained scheduling tables. The approach also significantly outperforms the fully distributed

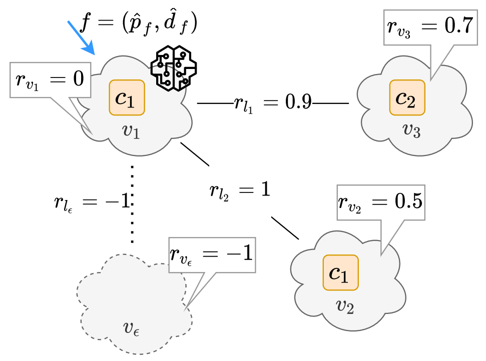


Fig. 5: The DRL agent at each node locally observes and controls individual flows. (Figure from [35]; ©2021 IEEE.)

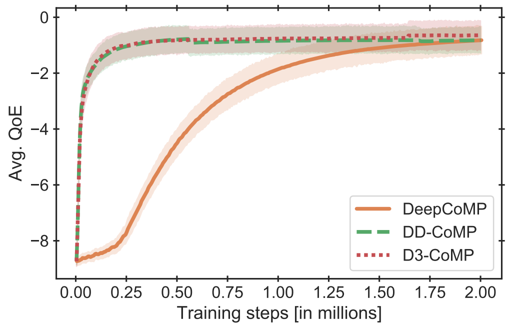


Fig. 6: Distributed DD-CoMP and D3-CoMP learn much faster than centralized DeepCoMP. (Figure adapted from [36].)

heuristics of Sec. IV-C as it can self-adapt to varying traffic patterns, services, and networks.

### D. *Self-Learning Coordination for Mobile Networks [36]*

Finally, I study how to apply self-learning DRL approaches to coordination in wireless mobile networks. I propose three novel DRL approaches, DeepCoMP, DD-CoMP, and D3-CoMP, for controlling which mobile users to serve by how many and which cells using coordinated multipoint (CoMP) [36]. DeepCoMP leverages central observations and control of all users to select cells almost optimally. In turn, DD-CoMP and D3-CoMP are multi-agent DRL approaches, that learn much faster than DeepCoMP (Fig. 6), allowing distributed, robust, and highly scalable coordination. All three approaches learn from experience and self-adapt to varying scenarios, supporting online transfer learning, and reaching 2x higher Quality of Experience (QoE) than existing approaches. Compared to related approaches, they have very few built-in assumptions and do not need prior system knowledge, making them more robust to change and better applicable in practice. The corresponding simulation environment, mobile-env [37], is available as open source, allowing others to easily train their own DRL approaches and build upon my work.

## VI. Conclusions and Impact

I propose several approaches for network and service coordination, each with different strengths and weaknesses. The conventional coordination approaches in Sec. IV are most suitable for well-understood and rather static scenarios, where they can build on existing expert knowledge. In these cases, they can provide reliable and close-to-optimal performance. Here, the centralized BSP approach (Sec. IV-A) can jointly coordinate scaling, placement, scheduling, and routing in scenarios with globally available traffic information and control. The fully distributed heuristics (Sec. IV-C) focus on large-scale scenarios with many, rapidly arriving flows, where they only need local information and benefit from their simplicity and speed. The hierarchical approach (Sec. IV-B) presents an intermediate solution between highly optimized centralized approaches and highly scalable distributed approaches.

While conventional coordination approaches still constitute the large majority of existing approaches, a clear trend goes towards machine learning approaches (Sec. V). These approaches rely less on predefined rules or human expert knowledge and instead leverage available data to *learn* network and service coordination. They can either support existing conventional approaches (Sec. V-A) or even replace them completely (Sec. V-B to V-D). While the approach of Sec. V-A can be trained on automatically generated VNF benchmarking data, the DRL approaches of Sec. V-B to V-D are trained in interaction with a (simulated) network environment through self-learning. Their self-learning and self-adaption capabilities make them ideal for many practical scenarios, which are not perfectly understood and may change dynamically over time. Again, the proposed approaches in Sec. V-B to V-D range from centralized to distributed architectures, which are useful for different problem sizes, traffic patterns, and compute resources. The distributed DRL approaches of Sec. V-C and V-D are a good choice for most scenarios since they combine my insights and lessons learned from all previous approaches. As such, they offer very good solution quality, self-adapt to various scenarios, and are fast and scalable.

My contributions focus on the conceptual and algorithmic problem of network and service coordination, which is fundamental in many real-world use cases, such as NFV, SDN, cloud and edge computing, 5G and 6G, and even for distributed machine learning pipelines [2]–[5]. Here, my approaches could be implemented in NFV MANO systems, SDN controllers, orchestrators like Kubernetes, or improve CoMP in 5G or 6G systems. All of my approaches are publicly available as open source (see links in the corresponding papers) and have been contributed to various projects such as the European H2020 5GTANGO project or collaborations with Huawei Germany. While industry adoptions still requires technical implementation, integration, and field studies, I do believe that my contributions can have significant impact and improve considerably over existing work. In fact, they can lead to better user experience, lower costs, and reduced environmental footprint, making them increasingly relevant in the future.

## Acknowledgments

This work was supported in part by the European Union's Horizon 2020 research and innovation programme under grant H2020-ICT-2016-2 761493 (5GTANGO), the German Research Foundation within the Collaborative Research Centre "On-The-Fly Computing" (SFB 901) and the German Federal Ministry of Education and Research through Software Campus grant 01IS17046 (RealVNF).

## References

[1] S. Schneider, "Network and service coordination: Conventional and machine learning approaches," Ph.D. dissertation, Paderborn University, Germany, 2022, https://digital.ub.uni-paderborn.de/hs/6209882.

[2] J. G. Herrera and J. F. Botero, "Resource allocation in NFV: A comprehensive survey," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 518–532, 2016.

[3] F. Z. Yousaf, M. Bredel, S. Schaller, and F. Schneider, "NFV and SDN—key technology enablers for 5G networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, 2017.

[4] C.-H. Hong and B. Varghese, "Resource management in fog/edge computing: A survey on architectures, infrastructure, and algorithms," *ACM Computing Survey*, vol. 52, no. 5, Sep. 2019.

[5] ITU-T, "Architectural framework for machine learning in future networks including IMT-2020 (Y.3172)," 2019.

[6] L. Popa, G. Kumar, M. Chowdhury, A. Krishnamurthy, S. Ratnasamy, and I. Stoica, "FairCloud: Sharing the network in cloud computing," in *ACM SIGCOMM Conference*, 2012, pp. 187–198.

[7] K. Kaur, V. Mangat, and K. Kumar, "A comprehensive survey of service function chain provisioning approaches in SDN and NFV architecture," *Computer Science Review*, vol. 38, 2020.

[8] H. Moens and F. De Turck, "VNF-P: A model for efficient placement of virtualized network functions," in *International Conference on Network and Service Management (CNSM)*. IEEE, 2014, pp. 418–423.

[9] F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and O. C. M. B. Duarte, "Orchestrating virtualized network functions," *IEEE Transactions on Network and Service Management*, vol. 13, no. 4, pp. 725–739, 2016.

[10] S. Ahvar, H. P. Phyu, S. M. Buddhacharya, E. Ahvar, N. Crespi, and R. Glitho, "CCVP: Cost-efficient centrality-based VNF placement and chaining algorithm for network service provisioning," in *IEEE Conference on Network Softwarization (NetSoft)*. IEEE, 2017, pp. 1–9.

[11] T.-W. Kuo, B.-H. Liou, K. C.-J. Lin, and M.-J. Tsai, "Deploying chains of virtual network functions: On the relation between link and server usage," *IEEE/ACM Transactions On Networking*, vol. 26, no. 4, pp. 1562–1576, 2018.

[12] F. Samuel, M. Chowdhury, and R. Boutaba, "PolyViNE: policy-based virtual network embedding across multiple domains," *Journal of Internet Services and Applications*, vol. 4, no. 1, pp. 1–23, 2013.

[13] T. Ghazar and N. Samaan, "Hierarchical approach for efficient virtual network embedding based on exact subgraph matching," in *IEEE Global Telecommunications Conference (GLOBECOM)*. IEEE, 2011, pp. 1–6.

[14] I. Houidi, W. Louati, and D. Zeghlache, "A distributed virtual network mapping algorithm," in *IEEE International Conference on Communications (ICC)*. IEEE, 2008, pp. 5634–5640.

[15] M. T. Beck, A. Fischer, H. de Meer, J. F. Botero, and X. Hesselbach, "A distributed, parallel, and generic virtual network embedding framework," in *IEEE International Conference on Communications (ICC)*. IEEE, 2013, pp. 3471–3475.

[16] R. Boutaba, M. A. Salahuddin, N. Limam, S. Ayoubi, N. Shahriar, F. Estrada-Solano, and O. M. Caicedo, "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities," *Journal of Internet Services and Applications*, vol. 9, no. 1, pp. 1–99, 2018.

[17] X. Zhang, C. Wu, Z. Li, and F. C. Lau, "Proactive vnf provisioning with multi-timescale cloud resources: Fusing online learning and online optimization," in *IEEE Conference on Computer Communications (INFOCOMM)*. IEEE, 2017, pp. 1–9.

[18] T.-H. Lei, Y.-T. Hsu, I.-C. Wang, and C. H.-P. Wen, "Deploying QoS-assured service function chains with stochastic prediction models on VNF latency," in *IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*. IEEE, 2017, pp. 1–6.

[19] S. Van Rossem, W. Tavernier, D. Colle, M. Pickavet, and P. Demeester, "Profile-based resource allocation for virtualized network functions," *IEEE Transactions on Network and Service Management*, vol. 16, no. 4, pp. 1374–1388, 2019.

[20] N. C. Luong, D. T. Hoang, S. Gong, D. Niyato, P. Wang, Y.-C. Liang, and D. I. Kim, "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3133–3174, 2019.

[21] R. Solozabal, J. Ceberio, A. Sanchoyerto, L. Zabala, B. Blanco, and F. Liberal, "Virtual network function placement optimization with deep reinforcement learning," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 2, pp. 292–303, 2019.

[22] L. Gu, D. Zeng, W. Li, S. Guo, A. Y. Zomaya, and H. Jin, "Intelligent vnf orchestration and flow scheduling via model-assisted deep reinforcement learning," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 2, pp. 279–291, 2019.

[23] Z. Xu, J. Tang, J. Meng, W. Zhang, Y. Wang, C. H. Liu, and D. Yang, "Experience-driven networking: A deep reinforcement learning based approach," in *IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2018, pp. 1871–1879.

[24] Y. S. Nasir and D. Guo, "Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, pp. 2239–2250, 2019.

[25] J. A. Ayala-Romero, A. Garcia-Saavedra, M. Gramaglia, X. Costa-Perez, A. Banchs, and J. J. Alcaraz, "vrAIn: A deep learning approach tailoring computing and radio resources in virtualized RANs," in *ACM Conference on Mobile Computing and Networking (MobiCom)*, 2019, pp. 1–16.

[26] J. Halpern and C. Pignataro, "Service function chaining (sfc) architecture," https://tools.ietf.org/html/draft-ietf-sfc-nsh-04, Tech. Rep., 2015.

[27] Cloud Native Computing Foundation, "Kubernetes: Production-grade container orchestration," https://kubernetes.io/.

[28] S. Dräxler, S. Schneider, and H. Karl, "Scaling and placing bidirectional services with stateful virtual and physical network functions," in *IEEE Conference on Network Softwarization (NetSoft)*, 2018, pp. 123–131.

[29] S. Schneider, M. Jürgens, and H. Karl, "Divide and conquer: Hierarchical network and service coordination," in *IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2021.

[30] S. Schneider, L. D. Klenner, and H. Karl, "Every node for itself: Fully distributed service coordination," in *International Conference on Network and Service Management (CNSM)*. IFIP/IEEE, 2020.

[31] S. Schneider, N. P. Satheeschandran, M. Peuster, and H. Karl, "Machine learning for dynamic resource allocation in network function virtualization," in *IEEE Conference on Network Softwarization (NetSoft)*, 2020.

[32] M. Peuster, S. Schneider, and H. Karl, "The softwarised network data zoo," in *International Conference on Network and Service Management (CNSM)*. IEEE, 2019, pp. 1–5.

[33] S. Schneider, A. Manzoor, H. Qarawlus, R. Schellenberg, H. Karl, R. Khalili, and A. Hecker, "Self-driving network and service coordination using deep reinforcement learning," in *International Conference on Network and Service Management (CNSM)*. IFIP/IEEE, 2020.

[34] S. Schneider, R. Khalili, A. Manzoor, H. Qarawlus, R. Schellenberg, H. Karl, and A. Hecker, "Self-learning multi-objective service coordination using deep reinforcement learning," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 3829–3842, 2021.

[35] S. Schneider, H. Qarawlus, and H. Karl, "Distributed online service coordination using deep reinforcement learning," in *IEEE International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2021.

[36] S. Schneider, H. Karl, R. Khalili, and A. Hecker, "DeepCoMP: Coordinated multipoint using multi-agent deep reinforcement learning," *Under Review. Preprint: https://ris.uni-paderborn.de/download/33854/33855*, 2023.

[37] S. Schneider, S. Werner, R. Khalili, A. Hecker, and H. Karl, "mobile-env: An open platform for reinforcement learning in wireless mobile networks," in *Network Operations and Management Symposium (NOMS)*. IEEE/IFIP, 2022.