Morphable Networks for Cross-layer and Cross-domain Programmability

Morphable Networks for Cross-layer and Cross-domain Programmability

C. F. Chiasserini*, S. Bizzarri†, C. E. Costa‡, G. Davoli§, J. Llorca¶, V. L. Lucrezia[xii], F. Malandrino‖, S. Miano**, A. Molinaro††, S. Palazzo‡‡, F. Risso*, D. Ronzani[x], S. Salsano[xi], G. Verticale**
* Politecnico di Torino † TIM ‡ CNIT § University of Bologna ¶ University of Naples Federico II ‖ CNR
** Politecnico di Milano †† University of Reggio Calabria ‡‡ University of Catania [x] Athonet/HPE
[xi] University of Roma Tor Vergata [xii] TIESSE

***Abstract*—Next-generation networks are expected to adapt not only to a wide range of application demands, but also to the specific needs of individual users, end device heterogeneity, and changing operational conditions. In this work, we discuss how to enhance the *programmability* of network devices (e.g., radio end-devices, base stations, routers) to achieve an unprecedented level of network adaptation and customization in 6G systems. We focus on two main innovations: *full network programmability* and *morphable networking*. The former breaks down traditional boundaries between protocol layers as well as domains (e.g., network segments, technological realms, administrative domains); the latter enables the protocol stack of each network node to be dynamically and consistently reconfigured across different domains, enabled by embedded artificial intelligence.**

I. Introduction

6G networks will need to deal with a highly intricate ecosystem, encompassing a diverse array of demanding stakeholders. As depicted in Fig. I, such an ecosystem may include challenging applications (e.g., extended reality, smart factories, and autonomous transportation), human users with their specific habits and preferences, terrestrial and non-terrestrial operational environments, as well as service providers and specialized verticals, each with its own target key performance and value indicators.

To enable the requirements of such a complex ecosystem, 6G networks have to dramatically improve their level of *customization and dynamically adapt* to:

- the needs of individual users, verticals, infrastructure owners, application providers, as well as overall context and operating environment;
- network characteristics, existing resources, and operational conditions, e.g., available radio, core, and transport technologies and their performance;
- the network hardware and its computing and connectivity capabilities;
- the nature of emerging services and applications, with their increasingly distributed structure and strict performance and isolation requirements.

Although present-day programmable network hardware offers a reasonable level of flexibility, its adoption alone does not enable full end-to-end network customization. Accordingly, in this paper we lay the foundations of a novel network paradigm, referred to as *morphable networks*. The high-level purpose of the proposed network paradigm is to extend programmability across the protocol stack as well as network domains

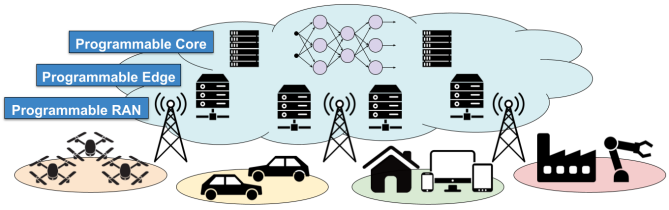


Fig. 1. Representation of the complex ecosystem, with heterogeneous network tiers (core, edge, radio access network (RAN)) and domains, that 6G networks are expected to address.

(network tiers such as far edge or edge, technological realms, and administrative domains) to achieve the required level of customization and performance.

Indeed, present-day networks target varying levels of either vertical programmability (i.e., across protocol layers in a single domain) or horizontal programmability (i.e., across domains for a single layer). We envision combining both paradigms, thus mixing and matching network capabilities as needed, crossing both layers and domain boundaries. This approach enhances the flexibility required from network devices to efficiently accommodate a wide range of hyper-customized services and applications over a common physical infrastructure.

To this end, we introduce the concept of *morphable network nodes*, where the protocols and functions to be run by each communication node and across different domains can be dynamically configured and coordinated to match end-to-end service level objectives. Artificial Intelligence (AI) and Machine Learning (ML) algorithms running on network nodes will help dynamically select protocols, functions, and configuration settings to be adopted at each node. Furthermore, control and data plane protocols and functions can be implemented through custom programs as microservices, deployed within morphable nodes as needed.

These innovations entail an increase in network complexity, and governing it requires informed, swift decisions. Accordingly, we propose an architectural design for morphable networks, addressing the following intertwined issues: (*i*) which management entities should make which decisions and at which scale, (*ii*) how should management decisions be coordinated for consistent end-to-end service provisioning, and (*iii*) which algorithmic approaches are the most appropriate at each decision-making level. On this regard, a critical aspect is

to find the best trade-off between decision effectiveness, which requires more data and/or time, and decision efficiency, which requires reducing both.

On the other hand, the ability of morphable networks to adapt to diverse operational environments and applications will enable the creation of makeshift connectivity for swift recovery in the case of disasters, or on-the-fly network slices matching challenging requirements even in the case of scarce resource availability, as well as the seamless integration of heterogeneous nodes, such as 5/6G, IoT devices, UAVs, and even satellites. Furthermore, thanks to their ability to swiftly and autonomously reconfigure, morphable networks can easily support reactive techniques for increased network security, as they can disguise, hence counteract, attacks by continuously and strategically adding unpredictability and randomness to the configuration of the communication network.

The rest of the paper is organized as follows. Sec. II discusses some relevant related work and prevailing challenges around network programmability. Sec. III introduces the concept of morphable networks, while Sec. IV presents their architecture. Then challenges and opportunities offered by our vision are set forth in Sec. V. Finally, Sec. VI draws our conclusions.

II. Network Programmability

Network programmability has been a focal point of research for many years, primarily due to the challenges arising from the rigid structure of traditional network layers and domains. The concept of "programmable" networks has emerged as a possible solution to overcome the aforementioned lack of flexibility, enabling the embedding of custom code within various protocol layers, thereby enhancing flexibility and adaptability. This concept of programmability has expanded throughout network tiers (including the far edge, near edge, and metro data centers), across technological domains (such as radio and core), and through all layers of the protocol stack. It enabled the deployment of custom network services at various layers, including the application layer [1], [2], the transport layer [3], and the network layer with, e.g., unique routing protocol extensions [4]. Further, it has expanded into the physical layer, fostering programmability in optical and wireless systems.

This progression in network programmability can be categorized into two distinct approaches based on how it integrates within the network: *vertical or cross-layer, and* horizontal or cross-domain.

Vertical programmability refers to the ability to program functionality across different layers of a protocol stack, as represented in Fig. 2(a). In this context, a substantial body of work has focused on leveraging softwarization technologies to virtualize and disaggregate network functions, implemented as flexible interacting microservices at various layers of the protocol stack. A relevant example is [5], which focuses on the upper layers of the protocol stack and foresees network services as composed of elementary network functions. In the wireless domain, cross-layer interaction goes often down to the physical layer and can be realized by the interplay between Software Defined Network (SDN), concerning the upper layers, and Software Defined Radio (SDR) technologies, dealing with physical communication functions.

Horizontal programmability, i.e., programmability across different network domains, aims to achieve a higher level of end-to-end integration and automation, enabling a unified management view across traditionally isolated domains (see Fig. 2(b)). Essential to this concept is the standardization of interfaces and protocols for cross-domain interactions, ensuring that changes or policies implemented in one domain can be automatically understood and accommodated by others. This is particularly important for the efficient orchestration of disaggregated services and applications that span over the device-edge-cloud continuum. For example, open-source hardware and software are used in [6] to implement an open multi-access network platform, and in [7] to optimize the performance of services spanning across cloud-edge infrastructures, in both cases demonstrating the benefits of using standardized interfaces. Indeed, it is only through this coordination that end-to-end services can be provisioned over a heterogeneous infrastructure, while meeting the goals imposed by global policies. To this end, projects such as ACROSS [8] have introduced an architecture for end-to-end service orchestration over multiple heterogeneous domains, with a hierarchical structure, in which a logically-centralized orchestrator manages domain-level orchestrators.

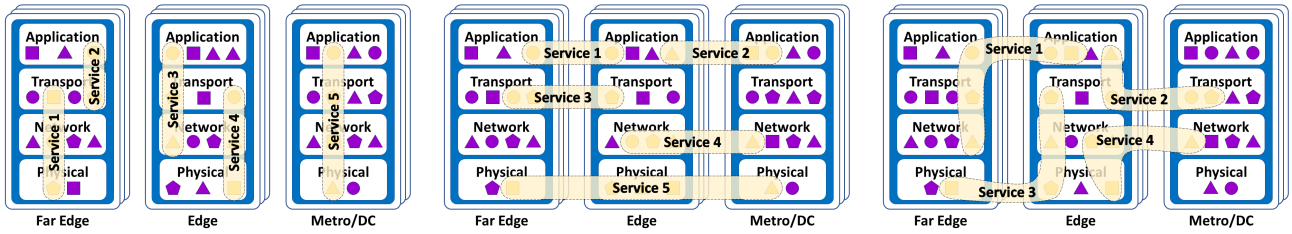
A. Full programmability

Our primary goal is to *go beyond vertical and horizontal programmability and to enable an integrated network-compute system with full programmability*, as shown in Fig. 2(c). The notion of full programmability tackles not just a single domain, node, network layer, or function. Rather, it represents a holistic approach that allows the deployment of functions across the "inter-domain stack", for true end-to-end control. Such concept of full programmability allows the deployment of custom programs, which, by being compiled automatically, enable the distribution of required functionality across the programmable multi-dimensional (vertical and horizontal) continuum. This end-to-end deep programmability transforms the network into a *highly adaptable platform* to meet the specific needs of network owners, users, and applications. As detailed in Sec. III and Sec. IV, our objective is thus to realize such a concept by designing a flexible, efficient, and intelligent network system.

B. Challenges to achieve full programmability

The concept of full programmability represents a significant advance in network management and operation. However, to maximize its potential, several challenges still need to be addressed.

Enhanced flexibility "across the stack." SDN has marked a significant step towards realizing end-to-end network programmability by opening control-plane APIs and allowing network owners to program their network directly using such interfaces. Nevertheless, current SDN applications still rely on a centralized controller [9], which forces network owners to a continuous interaction with such controller for deploying



(a) Vertical programmability (b) Horizontal programmability (c) Full programmability

Fig. 2. Conceptual representation of network programmability approaches. Different shapes represent different, generic functions. Shaded areas denote functions that have been programmed in a coordinated fashion to ensure a synergic behavior.

new functionalities. This choice is far from ideal but it is often adopted due to the lack of flexibility at the lower layers of the network stack, thus underscoring the need for a more distributed, ductile approach across the stack.

Appropriate abstractions for network control. Creating the appropriate set of abstractions is critical for granting network owners complete control over the network. Recent trends [10] offer the possibility to specify network behavior through high-level intents, allowing the network to be programmed top-down. However, those works often rely on an abstracted view of the underlying network, which can obscure the detailed understanding of the network behavior, making troubleshooting and optimization challenging. Thus, a balance needs to be found between high-level control and detailed visibility.

Fine-grained network verification. Current network verification methods, including static verification of the configuration of virtual radio access networks and routers, and the analysis of distributed protocols, often rely on abstract models and lack real-time insights on the network's actual state, especially during reconfiguration. With comprehensive visibility and deep programmability across the entire network stack, fine-grained telemetry data can be collected for each packet, enabling runtime verification of the network behavior.

Enabling dynamic network specialization. A fully programmable network environment gives the possibility of specializing network code based on real-time data and configurations. Such a dynamic specialization, based on live traffic patterns and network conditions, can lead to more efficient and responsive network operations.

Integrating AI/ML for advanced network orchestration. The application of AI and, in particular, ML techniques in network management and resource orchestration can significantly enhance decision-making in complex, data-intensive scenarios. To fully leverage AI/ML approaches, a network should be equipped with capabilities for deep observation and control at every level. This would enable AI/ML to move beyond generic decision-making to more precise, context-aware actions.

Facing the above challenges requires reevaluating and redesigning the network architecture and management protocols. The ultimate goal is to make networks evolve into deeply programmable entities, capable of adapting and responding dynamically to the unique demands and operational conditions as defined by network owners and operators as well as by the service and applications they have to support.

III. Morphable Programmable Networks

To address the above challenges in the design and development of programmable networks, we leverage and expand on current virtualization and programmability advances, and introduce a new concept, envisioning full – horizontal *and* vertical – programmability of the network system. We name such a concept *Morphable Programmable Networks (MPN)*, as the nodes composing the network will be capable of being reprogrammed from the physical layer up to the application layer.

More specifically, at every layer, an MPN node hosts several types of functions/protocols (e.g., a different MAC scheme at layer 2, or routing scheme at layer 3) implemented through, e.g., containerized microservices or low-level code logic (e.g., unikernels and eBPF) that can be enacted according to the specific requirements of the service and application to be supported, the characteristics of the surrounding environment, and the capability of the other nodes with which the tagged node has to interact. In doing so, the MPN represents a *reconfigurable network and computing platform* that is highly flexible and can fully adapt to the heterogeneity of both the operational context and the devices that need to communicate, or cooperatively collect and process data.

At the heart of the MPN concept lies a distributed, yet coordinated, management plane, which tailors the nodes' protocol stack to long-term dynamics of the environment and the network and computing system. Fully-programmable control and data planes are instead responsible for real-time adaptation to changing conditions, thus addressing the challenge concerning dynamic network specialization. We further detail the management, control, and data planes of an MPN below.

A. Management plane

In MPNs, through management plane functionalities, each node can:

- Gather requirements from applications and users, along with information about the context, operational conditions, and capabilities of other nodes from the surrounding environment.
- Select the most suitable configuration of each protocol layer for (*i*) ensuring a consistent, efficient protocol stack within individual nodes as well as across multiple nodes and network domains, (*ii*) matching the application requirements and environment characteristics, and (*iii*)

enabling the communication with the neighboring nodes with which interaction is necessary.
- Manage the lifecycle of the deployed functions and protocols (e.g., turning them on/off), at each single node, in a coordinated manner with the other nodes.

We remark that the two latter abilities are essential for creating a consistent protocol stack, where layers are not only compatible with each other but also synergically integrated and tailored to the specific context and network tasks.

To achieve the above objectives, each MPN node incorporates a *Morphing Engine (ME)*, equipped with the necessary intelligence to (*i*) decide on the node's protocol layers adaptations/configurations, possibly agreeing with the other nodes within the same administrative domain, and (*ii*) automatically trigger adaptations/configurations of the intra-node protocol stack, i.e., control and data planes, by starting the related in-node microservices. In addition, to enable a coordinated configuration of nodes belonging to different administrative domains, the MEs in a given domain can also elect a *Domain Morphing Engine (DME)*, in charge of interacting with similar logical entities in other domains. Similarly, MEs in different network tiers (cloud, edge, far-edge) can interact to coordinate the configuration of the nodes belonging to each tier.

B. Data plane

MPNs aim to enable *runtime programmability* into the data plane. Presently, data plane programmability mainly occurs during the pre-deployment phase, involving the customization of device behaviors through the compilation of new network programs and subsequent reconfiguration of the data plane before the device can start handling traffic. This process often requires the rerouting of traffic to isolate and reprogram devices before they are reintegrated into the network.

On the contrary, MPNs enable dynamic adaptability to real-time changes, allowing the data plane to optimize itself for current requirements and traffic workloads. This approach represents a significant difference from the static configurations seen in traditional networks. With MPNs, the data plane can directly and dynamically respond to time-varying requirements of the network, which is particularly beneficial for rapidly deploying new features. This will also allow offering immediate responses to increasingly insidious security threats, and integrating specific network customizations to enhance overall resource efficiency and performance (e.g., deploying tailored congestion control algorithms at hosts/SmartNICs for particular customers in cloud data centers).

However, achieving this vision presents substantial challenges. Indeed, to enable dynamic programming of the network, including the addition, modification, and removal of functions, new approaches to hardware programmability are needed. These include developing new hardware with advanced architectures like disaggregated Reconfigurable Match-Action Table (dRMT) [11] or employing *overlay-based* methods with a *SwitchVM* [12] on existing hardware platforms such as P4-based for runtime programmability. The key challenge here lies in finding a balance between these approaches, using existing hardware when possible and incorporating "lightweight overlays" to enhance real-time functionality.

We remark that, from a broad perspective, runtime data plane programming goes well beyond programming packet processing pipelines, like in P4. Vertically, elements such as host kernel stacks and SmartNICs, which use general-purpose programming models (like C or restricted C, e.g., eBPF), offer broader customization possibilities, including specialized congestion control and transport protocols. To enable a whole-network customization, it is required to develop new programming models and abstractions, coupled with a suitable Domain Specific Language (DSL) (e.g., eBPF or extensions thereof). These abstractions should effectively mask the complexity of vertical and horizontal distribution, as well as the heterogeneity in device architectures and performance characteristics. Ideally, a compiler should analyze the program, determine the most suitable component for each function, and distribute the program accordingly across the various network elements – a task that is even more challenging in a distributed network as an MPN.

C. Control plane

In MPNs, the role of the control plane is twofold. On one hand, it must guarantee that the network provides a reliable service that matches the application requirements even when the network changes because of events such as new user arrivals, unexpected traffic spikes, or sudden topology modifications. On the other hand, it is responsible for reading network events from the data plane and configuring the data plane consistently across multiple nodes. Whenever the data plane is reprogrammed, i.e., its tables and interfaces change, it is important that no information is lost in the morphing. Therefore, the control plane should be tightly integrated with the data plane, be able to communicate with heterogeneous and dynamic nodes, and preserve nodes' state whenever they are reprogrammed. As a consequence, in an open MPN, standardization will be mostly focused on the design of APIs and abstractions, even more than protocols.

From a programming point of view, control plane programmability has been one of the driving forces towards network disaggregation. Switch operating systems such as ONL (http://opennetlinux.org) or SONiC (https://sonicfoundation.dev) run each component as an independent software module or container that can be updated at runtime. Network owners do not directly program the control plane, but specify application and network requirements as high-level intents expressed in a declarative programming language, or even in natural language [10]. An intent compiler, possibly equipped with the necessary natural language processing tools, generates the corresponding control plane code, which is then deployed on the relevant nodes. According to the horizontal programming approach, the compiler is also responsible for generating the control protocol between the nodes. To do so, MPNs can exploit the availability, in switches and SmartNICs, of programmable hardware or specialized hardware such as neural-network inference engines (e.g., Broadcom Trident 5-X12). In a vertical programming approach, the intent compiler could determine which logic could be offloaded to the data plane, and generate the corresponding code and the protocol between

TABLE I
CHALLENGES ADDRESSED BY MPN

Challenges	MPN Enabler
Flexibility across the stack	On-demand, per-protocol layer microservice activation
Appropriate abstractions for network control	Intent compiler addressing heterogeneous infrastructure nodes
Fine-grained network verification	In-network computing for traffic analysis
Dynamic network specialization	Real-time programmable Data and Control planes
AI/ML for network orchestration	AI/ML-driven Management, Control and Data planes

the control plane and the data plane. In a full programmability approach, instead, the number of choices the intent compiler must make grows quickly, and the optimal choice depends on the likelihood of future traffic demand and failure patterns. Hence, the use of AI/ML becomes essential to identify the most likely scenarios and to predict the outcome of the different possible choices. Furthermore, AI/ML techniques can be used to enable an automatic verification that the running code correctly implements the intent as the network evolves over time.

Based on the above discussion, Table I summarizes how the MPN concept addresses the challenges highlighted in Sec. II-B.

IV. END-TO-END MANAGEMENT ARCHITECTURE

As mentioned, the management plane of an MPN spans over different technological and administrative domains, thus creating a *programmable network continuum*. MPN management – or, more precisely, *morphing* – is performed in a distributed way within each domain and across different ones. Within each domain, the ME at one of the nodes takes the logical role of DME, either through designation by the domain's administrator or through an election process. Indeed, the additional capabilities required for performing DME duties are present in all MEs, but are activated at an ME only upon taking the DME role. The DME then interacts with the DMEs of other domains to achieve network-wide service provisioning.

To realize the functionalities described in Sec. III, MEs and DMEs are assigned the tasks represented as functional elements in Fig. 3. Specifically, the proposed architecture performs end-to-end network management by intelligently scheduling the deployment and activation of functionalities at the different layers of the morphable nodes' programmable stack.

The DME is in charge of maintaining the catalog of services available in the domain (*Service catalog* block), which are consistent with the capabilities of the morphable nodes in the domain. Moreover, the DME is responsible for establishing policies for the programmability of the nodes within the domain (*Programmability policy* block), to help support the coordination between MEs, with the aim of achieving intra-domain consistent programmability. In general, policies established by DMEs may limit the set of protocols that can be used at any layer of the protocol stack, as well as impose requirements on their performance. Examples of such policies include enforcement of certain traffic delivery patterns, limiting the delay at each hop at the data link layer, or imposing requirements on the network and transport protocols. By doing so, DMEs define *protocol stack blueprints*, i.e., templates for each ME to use for the programming of the node it is responsible for. Once DMEs have set the blueprints, MEs adapt them based on the application and service requirements they need to satisfy.

This approach inherently facilitates the creation of clusters of nodes within a domain that performs a certain service, with some nodes possibly belonging to multiple clusters, as they can contribute to multiple services. In such nodes, the ME may need to activate more than one blueprint simultaneously at each protocol layer, and steer the traffic through the node protocol layers accordingly.

Additionally, the interactions among DMEs of different domains are key to realize end-to-end cross-domain service provisioning, which includes (see Fig. 3): the interaction between the system and external users (*User interface* block), the global view over the available services (*Service database* block), and all the operations necessary for automatic control and management of resources and services (*Zero-touch orchestration* block).

Regarding the ME acting locally at each network node in a domain, it receives the possible protocol stack blueprints from the DME and activates one of them based on (*i*) the information on the underlying local resources gathered through the monitoring process (*Monitoring* block), and (*ii*) a coordination process performed jointly with the other MEs in the domain on how to program the protocol stack in a concerted way (*Coordinated programmability* block). The ME then proceeds with the deployment of the functionalities at the different protocol layers (*Local node programmability* block), as agreed with its neighboring MEs.

The services offered across the different domains are exposed by the DMEs and can be requested by any external user, which can specify associated service level agreements. Importantly, users may also be (*i*) developers and administrators that may need to control and/or manage the morphable nodes and domains, (*ii*) external applications that may need specific network services from the MPN, and (*iii*) the DME of a different domain requesting the reconfiguration of a remote domain to complete the deployment of an end-to-end service.

Automatic operations within the MPN are aided by AI/ML algorithms, which are crucial in providing the system with adaptability to the inherent dynamicity of the infrastructure. In particular, AI/ML algorithms can be used to select the appropriate microservices to compose a given service, at which node and domain to place them, as well as the protocols and functionality to deploy within the individual nodes' protocol stack. Different AI/ML approaches will be required for service and system orchestration, depending upon the nodes'

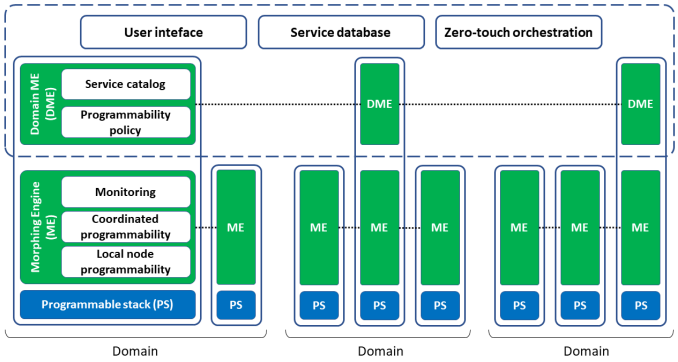


Fig. 3. Reference architecture of the MPN including the functional elements constituting the management entities at different levels (i.e., end-to-end management plane, DMEs, and MEs) and of their interactions.

characteristics and the timescale at which the decision-making process has to take place. In fact, morphing (i.e., system management) decisions and operations should be performed at a slower pace than those concerning control and data planes. Furthermore, the choice of the most appropriate algorithm may vary according to inherent features, including their scalability and energy consumption, as well as on external aspects, e.g., the availability of resources at the nodes and within a domain, or the intensity of service requests.

V. Research Challenges and Future Directions

The proposed concept of MPNs opens several research avenues; in this section, we discuss the most prominent ones.

Adaptive configuration of data and control plane. The concept of MPN brings to the forefront the intricate task of dynamically activating and configuring the data and control planes. This process, pivotal in responding to the ever-changing network demands, encompasses not only the efficient allocation of resources but also the adaptability to shifting traffic patterns and network conditions in real time.

The main challenge lies in developing a distributed system capable of making swift and accurate decisions and implementing them effectively without jeopardizing the network integrity or performance. From the viewpoint of decision-making schemes, correct decisions, even optimal ones, are useless if they come too late. This means that algorithm design must also account for (i) *which network entities* make the decisions, (ii) *when*, i.e., how frequently, in response to which events and/or by which target time, and (iii) *how*, i.e., based upon which information. The latter two points are deeply linked to the issues of network monitoring and the data flow within and across network domains. Intuitively, more data results in better decisions; on the negative side, they might generate overhead and delays at different locations within the network.

A related challenge is ensuring that requirements and decisions are properly communicated. To this end, we need to create Domain Specific Languages (DSLs) that can accurately describe the performance requirements of the various services and simplify the programming of heterogeneous nodes. However, the development of compilers capable of efficiently translating high-level abstractions into executable code while also optimizing the distribution of functions across a heterogeneous network environment remains an open issue.

AI/ML Models for Decision Making. Similar concerns arise when using AI/ML models, for both training and inference. During training, it is critical to estimate and account for the effect of both the quantity and the quality of the data being used; importantly, both are influenced by the location of the data source within the network. Such sources must not necessarily be the same used at inference time; indeed, *transfer learning* techniques allow training a model using data that is more plentiful and/or easier to obtain, and exploit it for other data at other locations at inference time. Furthermore, under paradigms like split learning, the computational resources of *multiple* network nodes can be exploited to perform a single learning or inference task. Accordingly, node-selection decisions need to account for such diverse aspects as network delays, computational capabilities, and data transfer policies.

Security and privacy threats. MPNs bring tremendous simplification to the management of security of large, business-critical, multi-tenant networks. This comes, however, with newer attack surfaces and challenges that must be addressed. First, MPNs are intrinsically multi-tenant, i.e., the same infrastructure is shared by multiple services with different security requirements, including e-health applications and safety-critical industrial protocols. Thus, the MPN data plane must ensure that the traffic and the resources of the tenants are isolated from each other. The study in [13] has made a first step in this direction, but more research is needed to support real-time programmability. In addition, any network carrying security and safety critical traffic must undergo extensive verification to provide trustworthiness. How to establish trustworthiness in real-time is an open issue, which may be addressed by using formal methods and safeguards [14]. Finally, extensive usage of ML paves the way to Adversarial Machine Learning attacks aiming, e.g., at steering traffic onto unsafe links or hijacking resources. Although progress has been made by using robust and explainable ML models [15], the mitigation of these attacks is still a critical challenge to be faced.

VI. Conclusion

Next-generation networks will need unprecedented levels of customization in order to support emerging services and applications; indeed, networks will need to customize their performance and behavior to the individual users and devices therein. This work has introduced our vision to enable such customization, focusing on two main avenues of innovation. The first is *full programmability* of the network, allowing decisions to be made across all protocol layers and administrative domains, using global information and pursuing global goals. The second is *morphable networking*, a network paradigm allowing to dynamically change the protocol stack running on each network node as needed. We also proposed the network architecture needed to support fully-programmable and morphable networking. Importantly, AI/ML techniques will play a significant role in end-to-end service orchestration and network management, as well as for the development of the morphable networks control and data plane. Finally, we discussed the challenges posed by fully-programmable and morphable networks, as well as presented relevant research directions towards the realization of morphable networks.

Acknowledgment

This work was partially supported by the EU under the NextGenerationEU partnership RESTART (PE00000001), through the SUPER and LEGGERO projects.

References

[1] S. Qi, L. Monis, Z. Zeng, I.-c. Wang, and K. K. Ramakrishnan, "SPRIGHT: Extracting the Server from Serverless Computing! High-Performance EBPF-Based Event-Driven, Shared-Memory Processing," in *ACM SIGCOMM*, New York, NY, USA, 2022, p. 780–794.

[2] S. G. Kulkarni, W. Zhang, J. Hwang, S. Rajagopalan, K. K. Ramakrishnan, T. Wood, M. Arumaithurai, and X. Fu, "NFVnice: Dynamic Backpressure and Scheduling for NFV Service Chains," *IEEE/ACM Transactions on Networking*, vol. 28, no. 2, pp. 639–652, 2020.

[3] M. Jadin, Q. De Coninck, L. Navarre, M. Schapira, and O. Bonaventure, "Leveraging eBPF to Make TCP Path-Aware," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2827–2838, 2022.

[4] T. Wirtgen, T. Rousseaux, Q. D. Coninck, N. Rybowski, R. Bush, L. Vanbever, A. Legay, and O. Bonaventure, "xBGP: Faster innovation in routing protocols," in *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, Boston, MA, Apr. 2023, pp. 575–592.

[5] H. Koumaras, D. Tsolkas, J. Garcia, D. Artunedo, B. Garcia, R. Marco, A. Salkintzis, D. Fragkos, G. Makropoulos, F. Setaki *et al.*, "A network programmability framework for vertical applications in the beyond 5g era," in *2022 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*. IEEE, 2022, pp. 375–380.

[6] T. Tachibana, K. Sawada, H. Fujii, R. Maruyama, T. Yamada, M. Fujii, and T. Fukuda, "Open multi-access network platform with dynamic task offloading and intelligent resource monitoring," *IEEE Communications Magazine*, vol. 60, no. 8, pp. 52–58, 2022.

[7] M. Gharbaoui, C. Contoli, G. Davoli, D. Borsatti, G. Cuffaro, F. Paganelli, W. Cerroni, P. Cappanera, and B. Martini, "An experimental study on latency-aware and self-adaptive service chaining orchestration in distributed NFV and SDN infrastructures," *Computer Networks*, vol. 208, p. 108880, 2022.

[8] D. Giannopoulos, G. Katsikas, K. Trantzas, D. Klonidis, C. Tranoris, S. Denazis, L. Gifre, R. Vilalta, P. Alemany, R. Muñoz *et al.*, "ACROSS: Automated zero-touch cross-layer provisioning framework for 5G and beyond vertical services," in *2023 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*. IEEE, 2023, pp. 735–740.

[9] Z. A. Bhuiyan, S. Islam, M. M. Islam, A. A. Ullah, F. Naz, and M. S. Rahman, "On the (in) Security of the Control Plane of SDN Architecture: A Survey," *IEEE Access*, 2023.

[10] A. Leivadeas and M. Falkner, "A survey on intent-based networking," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 625–655, 2023.

[11] J. Xing, K.-F. Hsu, M. Kadosh, A. Lo, Y. Piasetzky, A. Krishnamurthy, and A. Chen, "Runtime programmable switches," in *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, Renton, WA, Apr. 2022, pp. 651–665.

[12] S. Khashab, A. Rashelbach, and M. Silberstein, "Multitenant In-Network Acceleration with SwitchVM," in *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2024.

[13] R. Stoyanov and N. Zilberman, "MTPSA: Multi-Tenant programmable switches," in *ACM P4 Workshop in Europe (EuroP4)*, Dec. 2020, pp. 43–48.

[14] T. Gurriet, M. L. Mote, A. Ames, and E. Feron, "Establishing trust in remotely reprogrammable systems," in *ACM International Conference on Human-Computer Interaction in Aerospace (HCI-Aero)*, Sep. 2016, pp. 1–4.

[15] A. Carlevaro, S. Narteni, F. Dabbene, M. Muselli, and M. Mongelli, "Confiderai: a novel conformal interpretable-by-design score function for explainable and reliable artificial intelligence," 2023. [Online]. Available: https://arxiv.org/abs/2309.01778

Carla Fabiana Chiasserini Carla Fabiana Chiasserini (F'18) was a visiting researcher at UCSD and as a Visiting Professor at Monash University (2012, 2016), TUB (Germany) (2021-2022), and HPI, Potsdam University (2023). She is currently a Professor at Politecnico di Torino and a WASP Guest Professor at Chalmers University.

Simone Bizzarri Simone Bizzarri is responsible for the definition and implementation of guidelines for the Network Management System (for fixed and mobile networks) in TIM in order to meet the requirements for ubiquitous automation, flexibility, integrability and openness to other systems in the management processes. He is also 3GPP SA5 delegate aiming to contribute to SON, AI/ML, energy efficiency and mobile network automation topics for 5G and 6G systems.

Cristina Emilia Costa is a Research Scientist at Consorzio Nazionale Interuniversitario per le Telecomunicazioni (CNIT), Italy. Her research interests are in energy efficient networks and multimedia applications.

Gianluca Davoli is an Assistant Professor (fixed-term) at the University of Bologna, Italy, where he obtained his Ph.D. in 2021. His main research interests are in architectures and algorithms for the orchestration of services across distributed, dynamic network infrastructures.

Jaime Llorca is an Assistant Professor at the University of Naples Federico II and a Research Professor at New York University. He previously served as a Senior Research Scientist at Bell Labs, New Jersey, and obtained his Ph.D. at the University of Maryland, College Park. His current research interests include next-generation networks, distributed edge/cloud computing, end-to-end service orchestration, and network AI.

Vincenzo Luciano Lucrezia obtained a degree in Computer Science from the University of Pisa. He worked in ENI, Olivetti, Italtel, and Infostrada and took an active part in the definition and development of local and geographic networks. Since 1998 he has been leading the technical side of Tiesse, creating a line of routers and M2M products.

Francesco Malandrino is a senior researcher at CNR-IEIIT. Prior to his current appointment, he was a fixed-term assistant professor at Politecnico di Torino, where he earner his PhD in 2012. His research interests include mobile networks and distributed machine learning.

Sebastiano Miano serves as an Assistant Professor (fixed-term) at the Politecnico di Milano, Italy. He received his Ph.D degree in Computer Engineering from Politecnico di Torino, Italy. His research interests focus primarily on the development of mechanisms and architectures to improve host-based network applications.

Antonella Molinaro is a professor of Telecommunications at the University Mediterranea of Reggio Calabria, Italy, and at University of Paris-Saclay, France. Her current research focuses on wireless and mobile networking, edge intelligence, future Internet.

Sergio Palazzo is a Professor of Telecommunication Networks at the University of Catania, Italy. His current research interests include wireless networks, machine learning in networking, network programmability, and protocols for the next generation of the Internet.

Fulvio Risso is Full Professor at Politecnico di Torino, Italy, where he got his Ph.D. in 2000. His research interests focus on network programmability, network functions virtualization, edge-to-cloud continuum and orchestration.

Daniele Ronzani is a Research Engineer at HPE, working in the Research & Innovation department. Before joining HPE, he worked in network communication research at University of Padua, Italy. He obtained his Ph.D degree in Brain, Mind and Computer Science in 2019.

Stefano Salsano is a professor at University of Rome Tor Vergata, Italy. His current research interests include network programmability, edge computing and network function virtualization

Giacomo Verticale received the Ph.D. degree in telecommunications engineering from the Politecnico di Milano, Italy, in 2003. He is currently an Associate Professor at the Politecnico di Milano. He was involved in several research projects on fixed and wireless broadband access technologies and promoting the smart grid. His current interests focus on the security issues of the smart grid, on network function virtualization, and on edge computing in 5G.