Morphable Networks For Cross-Layer And Cross-Domain Programmability: A Novel Network Paradigm

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 need to deal with a complex ecosystem, encompassing a plethora of demanding stakeholders. Such an ecosystem (Fig. I) includes challenging applications (e.g., XR, smart factories, and autonomous transportation), human users with their specific preferences, terrestrial/non-terrestrial environments, as well as service providers and specialized verticals, each with its own target key performance indicators.

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

- needs of individual users, verticals, infrastructure owners, and application providers;
- network characteristics, existing resources, and operational conditions;
- network hardware and its computing and connectivity capabilities;
- services' and applications' distributed structure and strict performance and isolation requirements.

A prominent approach to address these challenges is *network programmability* through, most notably, software-defined networking (SDN). In SDN, a *controller* program can be used to drive the behavior of network hardware without the need to configure the individual pieces of equipment. Although SDN-controlled programmable hardware endows networks with a reasonable level of flexibility, its adoption alone does not enable full end-to-end network customization. Specifically, the scope of the decisions to make and the information to leverage remain limited to a single domain (e.g., a wide area network) and/or one protocol layer. This is at odds with the structure of modern-day networks where emerging services span over multiple tiers, from mobile devices (the so-called "far-edge"), all the way to cloud servers (forming what is termed the device-edge-cloud continuum).

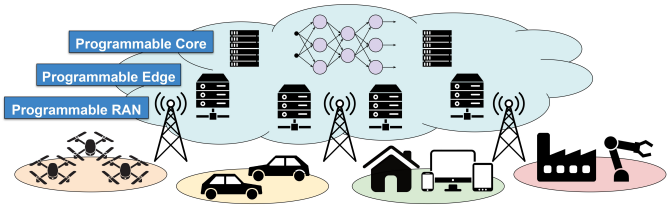
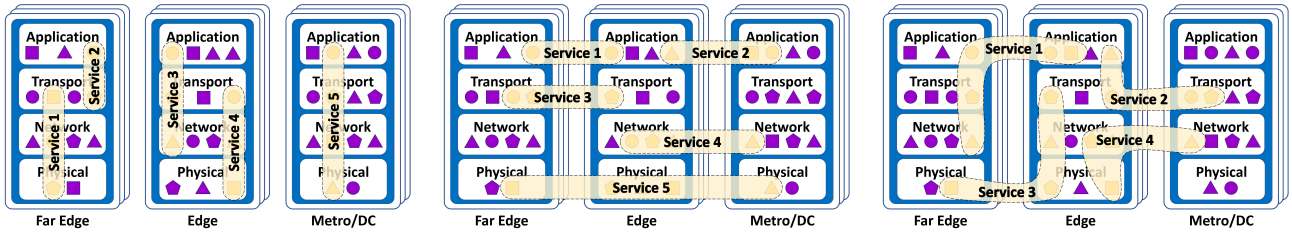


Fig. 1. Representation of the complex ecosystem, with heterogeneous network tiers (core, edge, radio access network (RAN)) and domains, that 6G networks are expected to address.

Accordingly, in this paper we lay the foundations of a novel network paradigm, referred to as *morphable networks* (MPNs). MPNs' purpose is to extend programmability across the protocol stack as well as network domains (network tiers, technological realms, and administrative domains) to achieve the required level of customization and performance. We envision combining the paradigms of vertical programmability (i.e., across protocol layers in a single domain) and horizontal programmability (i.e., across domains for a single layer), thus matching network capabilities as needed. This approach enhances the flexibility of network nodes, allowing them to accommodate a wide range of hyper-customized services and applications over a common physical infrastructure.

To this end, in MPNs the protocols and functions to be run by each node can be dynamically configured and coordinated to match end-to-end service level objectives. In line with self-driving and autonomous networking principles, Artificial Intelligence (AI) and Machine Learning (ML) algorithms running on network nodes will help dynamically and autonomously select protocols, functions, and configuration settings. To enable this, control and data plane protocols and functions can be implemented through custom programs as microservices, deployed within MPN 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 MPNs, 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



(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.

algorithmic approaches are the most appropriate at each decision-making level. On this regard, a critical aspect is trading off decision effectiveness, requiring more data and/or time, and decision efficiency, which requires reducing both.

On the positive side, the ability of MPNs to adapt to diverse operational environments and applications will enable creating makeshift connectivity for swift recovery, as well as on-the-fly network slices matching challenging requirements even in presence of scare resources or harsh operational conditions, as well as seamless integration of heterogeneous nodes (e.g., IoT devices, UAVs, and satellites). Of particular relevance is also the ability to create fully-customized, portable networks composed of personal user devices or Internet-of-Everything devices, which can reconfigure themselves to efficiently interact and optimally support immersive and cognitive digital applications for a personalized experience. With reference to the 5G-to-6G transition, it is worth remarking that the 3GPP working group (WG) SA1 has underlined the importance of native AI and network and computing coordination. In this context, the MPN paradigm can be considered as part of the future 6G because of its intrinsic flexibility, programmability, and adaptability characteristics. Finally, thanks to their ability to swiftly and autonomously reconfigure, MPNs can easily support reactive techniques for increased network security, as they can confound, hence counteract, attackers by continuously and strategically adding unpredictability and randomness to the configuration of the communication network.

In the following, we discuss related works and challenges around network programmability (Sec. II), and then introduce the MPN concept (Sec. III) and architecture (Sec. IV). Then, we highlight challenges and opportunities offered by MPNs (Sec. V) and draw our conclusions (Sec. VI).

II. Network Programmability

"Programmable" networks have emerged as a possible solution to overcome the aforementioned lack of flexibility, enabling embedded custom code within various protocol layers, [1]. This concept 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 protocol stack layers, enabling the deployment of custom network services at various layers (e.g., application [2], transport [3], network [4], and physical layer).

Efforts improving 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. Fig. 2 depicts these approaches using a simplified network stack, where layers represent general functionality (e.g., the network layer encompasses all routing operations) and do not refer to specific network protocols (e.g., IP).

Vertical programmability refers to the ability to program functionality across different protocol stack layers (Fig. 2(a)). In this context, several works have focused on virtualizing and disaggregating network functions into flexible interacting microservices at various protocol layers. As an example, [1], emphasizes the need for deep programmability spanning from high-level intents down through the control plane to the data plane, allowing for fine-grained control and verification at every level. In the wireless domain, the combination of SDN and software-defined radio is often used to jointly determine the modulation and coding scheme to be used on a link (a physical layer decision) and how much traffic shall be routed over that link (a network or transport layer decision).

Horizontal programmability refers to the ability to program functionality across different network domains (Fig. 2(b)). It allows a higher level of end-to-end integration and automation, enabling a unified management view across traditionally isolated domains. 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 accommodated by others. This is fundamental for the efficient orchestration of disaggregated services and applications (e.g., implemented through a set of interconnected containerized microservices) over the device-edge-cloud continuum. For example, open-source hardware and software can be used to implement an open multi-access network platform [5], and optimize service performance across cloud-edge infrastructures [6]. 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, prior work [7] has introduced an architecture for service orchestration over multiple heterogeneous domains, with a logically-centralized orchestrator managing domain-level orchestrators.

A. Full programmability

Our goal is to *go beyond vertical and horizontal programmability enabling an integrated network-compute system with full programmability* (Fig. 2(c)). This concept tackles not just a single domain, node, network layer, or function; rather,

it represents a holistic approach allowing the deployment of functions across the "inter-domain stack", for true end-to-end control. Full programmability allows the deployment of custom services, 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–Sec. IV, our objective is to realize such a concept by designing a flexible, efficient, and intelligent network system.

B. Challenges to achieve full programmability

Full programmability represents a significant advance in network management and operation however, it poses several challenges.

Enhanced flexibility "across the stack." SDN contributes to 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 [8], which engages network owners into a continuous interaction for deploying new functionalities. This is far from ideal but often adopted due to the lack of flexibility at the lower protocol layers, and a more ductile approach across the stack is needed.

Abstractions for network control. Creating an appropriate set of abstractions is critical for granting network owners complete network control. As an example, in a typical SDN scenario, nodes expose a flow table abstraction to program the forwarding tables independently from the technology, and at the network operating system layer, the controller exposes a big router abstraction to configure point-to-point connectivity. High-level intents [9] exploit abstractions to specify the network behavior in a top-down manner, thus simplifying network management and control. However, abstractions may hinder the detailed understanding of a network, making troubleshooting and optimization challenging.

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

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

Integrating AI/ML for network orchestration. AI/ML can significantly enhance network management and resource orchestration decision-making in complex, data-intensive scenarios. To fully leverage such approaches, a network needs to perform 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 redesigning the network architecture and management protocols. As discussed below (and summarized in Table I), we do so by envisioning that 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 services and applications they have to support.

III. Morphable Programmable Networks

To address the above challenges, we leverage and expand current virtualization and programmability techniques, and envision full – horizontal *and* vertical – programmability of the network system. We name such a concept *Morphable Programmable Networks*, as the nodes composing the network will be capable of being reprogrammed from the physical layer up to the application layer.

Specifically, at every layer, an MPN node hosts several types of functions/protocols (e.g., a different MAC scheme, or routing scheme) implemented through containerized microservices or low-level code logic (e.g., unikernels and eBPF) that can be enacted according to the 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. Thus, the MPN represents a *reconfigurable network and computing platform* that 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

Through management plane functionalities, each MPN node can:

- Gather information on applications' and users' requirements, context, operational conditions, and other nodes' capabilities.
- Select the best 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 communication with the neighboring nodes.
- Manage the lifecycle of the deployed functions and protocols (e.g., turning them on/off), in coordination with the other nodes.

Notably, the two latter abilities are essential for creating a consistent protocol stack, where layers are not only compatible

TABLE I
HOW MPNS ADDRESS THE CHALLENGES LISTED IN SEC. II-B

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

but also integrated with each other 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 intelligence required 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, by starting the related in-node microservices. Further, for coordinated configuration across administrative domains, the MEs in a given domain can elect a *Domain Morphing Engine (DME)*, responsible for interacting with similar logical entities in other domains.

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 starts handling traffic. This process often requires traffic rerouting 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. However, this vision presents substantial challenges. To enable dynamic network programming, including the addition/modification/removal of functions, new approaches to both software and hardware programmability are needed. These include developing new hardware with advanced architectures like disaggregated Reconfigurable Match-Action Table [10] or employing *overlay-based* methods with a *SwitchVM* [11] on existing hardware platforms such as P4-based for runtime programmability.

We remark that 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 (e.g., eBPF or extensions thereof). 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.

This combination of software and hardware programmability ensures that MPNs can meet the diverse and evolving requirements of next-generation network environments, as 5G networks [12] gradually evolve towards 6G. Indeed, runtime data-plane programming enables networks to dynamically adapt and easily deploy new 6G features like improved AI/ML and advanced resource management. Such adaptability is crucial for implementing new 3GPP standards, making the transition between network generations more manageable.

C. Control plane

The role of the control plane in MPNs is twofold. On one hand, it must ensure that the network provides a reliable service that matches the application requirements even upon events like 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, no information should be lost in the morphing. Therefore, the control plane should be tightly integrated with the data plane and preserve nodes' state whenever they are reprogrammed. Network owners do not directly program the control plane but specify application and network requirements as high-level intents expressed in a declarative programming or natural language. An intent compiler generates the corresponding control plane code, which is then deployed on the nodes. According to the horizontal-programming approach, the compiler should also generate the control protocol. To do so, MPNs can exploit the availability 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 can determine which logic to offload to the data plane, and generate the corresponding code and the protocol between control and 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 can be used for automatically verifying that the running code correctly implements the intent as the network evolves.

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

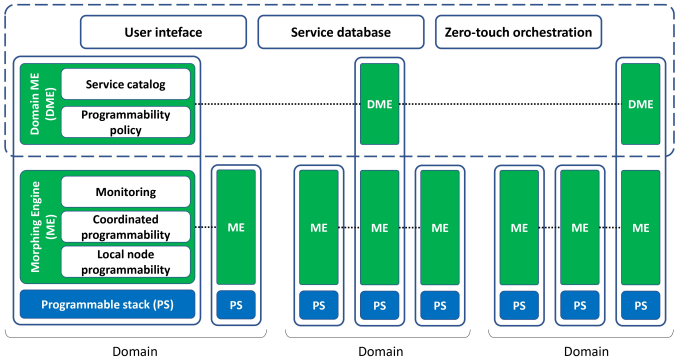


Fig. 3. Reference architecture of the MPN including the functional elements constituting the management entities at different levels (i.e., end-to-end management plane, DMEs, and MEs) and of their interactions.

IV. End-to-End Management Architecture

By spanning over different technological and administrative domains, the MPN management plane creates 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 domain ME (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.

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 node programmability within the domain (*Programmability policy* block), to better 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, and impose requirements on their performance: examples include enforcement of traffic delivery patterns, limiting the data-link delay, or imposing requirements on the network and transport protocols. DMEs thus 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 naturally facilitates the creation of clusters of nodes within a domain performing the same 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 between DMEs of different domains are key to realize 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. Among the ME's functionalities, the local node programmability block is essential in determining the optimal set of functions for different stack layers and enabling runtime optimization of these functions. This component will leverage dynamic compilation and optimization techniques [13], [14], adapting its behavior to real-time network metrics and optimizing it for specific traffic patterns, control plane policies, and flow rules. Periodic re-optimization ensures continuous performance improvement without disrupting ongoing network operations, thereby demonstrating the feasibility and potential of MPNs.

The services offered across 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 need to control and/or manage the morphable nodes and domains, (*ii*) external applications that 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 end-to-end services.

Automatic operations within the MPN are aided by AI/ML, which is crucial to select the appropriate microservices com-

posing a service, at which node and domain to place them, and the protocols and functionality to deploy within the nodes' protocol stack. Different AI/ML approaches are required for service and system orchestration, depending upon the nodes' characteristics, operational timescale and data availability.

Finally, we remark that the ME and DME can be implemented as programmable components by extending the concepts of near-Real time (RT) and non-RT RAN Intelligent Controllers (RIC) (resp.), introduced by the O-RAN specifications[1], or by extending the Management Function defined by the 3GPP TS 28.533. Each node can indeed be equipped with a DME entity akin to the non-RT RIC, which maintains a catalog of available services and defines the programmability policies that can be adopted (i.e., the protocol stack blueprints). Similarly, an ME entity akin to the near-RT RIC can implement the functions to be used at the different protocol layers as applications. Also, as per the O-RAN specifications, open interfaces can be defined, which support the interaction between the ME and the DME, as well as between the ME and the node protocol layers, and between the DME and external users.

Importantly, MPNs have a broader scope than O-RAN, which solely focuses on RAN programmability. Indeed, MPNs: (1) address virtually *all* layers of the protocol stack; (2) do not foresee any functional split, rather are designed for any type of node, including such low-power devices as wearables; (3) envision – as a critical feature–, that nodes deemed to communicate, agree on a protocol stack blueprint that can best adapt to the current context, supporting application requirements, and fulfilling needs and preferences of individual end users.

V. Research Challenges and Future Directions

As highlighted below, our proposed MPN concept opens several research avenues.

Adaptive configuration of data and control plane. A major challenge lies in making MPNs capable of swift and accurate decisions, and implementing them effectively without jeopardizing 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 network locations.

A related challenge is ensuring that requirements and decisions are properly expressed and communicated. To this end, we need to create Domain Specific Languages (DSLs) that can accurately describe the performance requirements of network 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. Finally, it is worth underlining the importance of developing such fully programmable data and control planes, along with the necessary interfaces, in such a way to maximize the contribution of MPNs to the development of fully programmable 6G networks – a topic currently under discussion within the 3GPP WG SA1.

AI/ML Models for Decision Making. 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* allows 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 dramatically simplify the security management of large, business-critical 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. [15] has made a first step in this direction, but more research is needed to support real-time programmability. Further, 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. Finally, extensive usage of ML paves the way to adversarial ML 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, the mitigation of these attacks is still a critical challenge to be faced.

VI. Conclusion

Next-generation networks need high customization to support emerging services and applications. This work has introduced our vision to enable such customization, focusing on two main innovation avenues: network *full programmability*, allowing decisions across all protocol layers and administrative domains, and *morphable networking*, a paradigm allowing for dynamic protocol stack adaptation. We also proposed the network architecture needed to realize the above vision, and underscored the relevance of AI/ML in end-to-end service orchestration and network management. Finally, we discussed the challenges posed by fully-programmable and morphable

[1]https://www.o-ran.org

networks, and presented relevant research directions towards their realization.

ACKNOWLEDGMENT

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

REFERENCES

[1] N. Foster, N. McKeown, J. Rexford, G. Parulkar, L. Peterson, and O. Sunay, "Using deep programmability to put network owners in control," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 50, no. 4, p. 82–88, oct 2020.

[2] 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.

[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] 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.

[6] 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.

[7] 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.

[8] 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.

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

[10] 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.

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

[12] M. Xie, A. J. Gonzalez, P. Grønsund, H. Lønsethagen, P. Waldemar, C. Tranoris, S. Denazis, and A. Elmokashfi, "Practically deploying multiple vertical services into 5g networks with network slicing," *IEEE Network*, vol. 36, no. 1, pp. 32–39, 2022.

[13] S. Miano, A. Sanaee, F. Risso, G. Rétvári, and G. Antichi, "Morpheus: A run time compiler and optimizer for software data planes," *IEEE/ACM Transactions on Networking*, vol. 32, no. 3, pp. 2269–2284, 2024.

[14] A. Farshin, T. Barbette, A. Roozbeh, G. Q. Maguire Jr., and D. Kostić, "Packetmill: toward per-core 100-gbps networking," in *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 1–17. [Online]. Available: https://doi.org/10.1145/3445814.3446724

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

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 earned 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.