Adaptive Risk-Aware Resource Orchestration for 5G Microservices over Multi-Tier Edge-Cloud Systems

Xingqi Wu†, Junaid Farooq† and Juntao Chen*
†Department of Electrical and Computer Engineering, University of Michigan-Dearborn, Dearborn, MI USA,
*Department of Computer and Information Science, Fordham University, New York, NY USA,
Emails: {xingqiwu, mjfarooq}@umich.edu, jchen504@fordham.edu.

***Abstract*—Modern fifth-generation (5G) networks are increasingly moving towards architectures characterized by softwarization and virtualization. This paper addresses the complexities and challenges in deploying applications and services in the emerging multi-tiered 5G network architecture, particularly in the context of microservices-based applications. These applications, characterized by their structure as directed graphs of interdependent functions, are sensitive to the deployment tiers and resource allocation strategies, which can result in performance degradation and susceptibility to failures. Additionally, the threat of deploying potentially malicious applications exacerbates resource allocation inefficiencies. To address these issues, we propose a novel optimization framework that incorporates a probabilistic approach for assessing the risk of malicious applications, leading to a more resilient resource allocation strategy. Our framework dynamically optimizes both computational and networking resources across various tiers, aiming to enhance key performance metrics such as latency, accuracy, and resource utilization. Through detailed simulations, we demonstrate that our framework not only satisfies strict performance requirements but also surpasses existing methods in efficiency and security.**

***Index Terms*—5G, microservices, edge-cloud infrastructure, resource orchestration.**

I. Introduction

The emergence of fifth-generation (5G) mobile networks presents a major revolution in the design and operation of modern communication and computing systems due to the increased softwarization and virtualization. Unlike its predecessors, 5G introduces a multi-tiered architectural paradigm that significantly diverges from the traditional, monolithic single-tier frameworks [1]. This transformation is not merely a technological evolution; it underpins a fundamental shift in how applications and services are designed, deployed, and managed. One of the key innovations enabling this paradigm shift is the widespread adoption of a microservices-based architecture for applications [2]. By breaking down complex applications into smaller, manageable, and independent units, microservices allow for greater modularity, easier scalability, and more effective fault isolation. While the architecture brings forth several benefits, it also imposes a set of unique challenges concerning the deployment and resource allocation of each microservice [3], [4]. Their performance, characterized by metrics such as latency and accuracy[1] , is highly dependent on both the tier they are deployed in and the amount of compute and networking resources allocated to them.

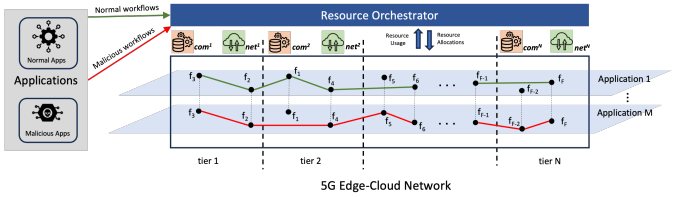


Fig. 1: Schematic representation of the resource orchestrator managing resources between normal and malicious applications considering the workflows of apps over the set of microservices f_1 through f_m.

However, this problem grows in complexity exponentially when considering a network hosting multiple applications, each comprising a unique directed graph of interconnected microservices. Orchestrating resources for these myriad services becomes a complex task since each microservice must be optimally placed within the appropriate tier of the 5G network and provisioned with the required amount of resources [5], [6]. Furthermore, an added layer of complexity arises from the necessity to consider the presence of potentially malicious applications. In the highly interconnected realm of 5G networks, the threat posed by deploying resources to rogue applications becomes a significant concern [7]. Such malicious applications have the potential to not only undermine the integrity of the network but also result in the gross misallocation of resources, thereby squandering essential compute and networking capabilities.

Resource management in cloud environments is a well-studied research problem [8]–[13]. However, the shift to microservices architecture brings new challenges, particularly in managing multiple application components across a vast number of distributed servers dynamically. In this context, the work in [2] models microservices allocation in cloud data centers as a binary quadratic programming problem, aiming to minimize interaction costs. A similar attempt to minimize application response time through workload profiling is discussed in [14]. With the rise of multi-access edge computing (MEC), researchers have explored application deployment and

This work was supported in part by the National Science Foundation (NSF) under Grant ITE-2226232.

[1]The notion of accuracy is used to reflect how well a computational task, e.g., object detection, is executed given the compute resources available.

migration in this setting as well [15]. For instance, [1] proposes a Markov decision process (MDP) formulation for dynamic service migration in MEC. The work in [16] adopts a machine learning approach for proactive placement and migration of microservices in MEC setups.

However, these works focus predominantly on either compute or network resources and often ignore the intricate coupling between the two. The authors in [17] have proposed an application based resource orchestration framework, however, they do not consider the possibility of malicious applications and the need for re-allocating resources. This paper tackles the overarching problem of optimizing resource allocation for microservices-based applications in a multi-tiered, heterogeneous 5G network environment. The main challenge is twofold: (i) To determine the optimal placement of individual functions across multiple network tiers, considering the real-time state of resources; (ii) To account for the coupling relationships between different resources like compute, storage, and bandwidth, and their impact on application-level metrics such as latency, accuracy, and cost. The key contributions of this paper are as follows:

1) We formalize the problem of function-to-tier mapping and resource allocation as a constrained optimization problem that takes into account both system performance and the potential of applications being malicious.
2) Our framework integrates a risk factor associated with the belief that an application might be malicious, thereby providing a more robust resource allocation strategy.
3) Through extensive simulations, we validate the effectiveness of our optimization framework in various scenarios, illustrating its adaptability and efficiency. We also provide comparisons with existing resource allocation strategies, demonstrating significant improvements in latency, accuracy, and resource utilization in the presence of risks.

The remainder of this paper is organized as follows: Section II provides a detailed overview of the system model. Section III presents the problem formulation and our optimization framework, followed by Section IV, which discusses the simulation setup and results. Finally, Section V concludes the paper and alludes to future research directions.

II. System Model

We consider a multi-tier edge-cloud 5G infrastructure, consisting of N tiers in the system, encompassing various types of compute nodes such as edge devices, MEC servers, and central cloud servers. The network architecture is assumed to host M applications, each represented as a directed acyclic graph (DAG). Our objective is to optimize the allocation of resources in such a way as to meet the application-specific performance metrics while minimizing resource utilization and considering the potential malicious nature of apps.

A. Physical Infrastructure and Computing Tiers

The physical infrastructure is constituted by a set $\mathcal{N}$ of computing tiers, with each tier $n \in \mathcal{N} := \{1, 2, \ldots, N\}$ having a fixed number of networking resources, denoted by net_n and compute resources, denoted by $comp_n$. These resources are shared across various applications that are inherently DAGs formed over a pool of functions or microservices. Fig. 1 illustrates an architecture of the 5G edge-cloud infrastructure with multiple computing tiers and applications along with fixed compute and network resources available at each tier. The resource orchestrator uses the application workflows and performance attributes to deploy and provision resources to the participating microservices.

B. Applications and Functions

We consider a universal set of functions or microservices, denoted by $\mathcal{F} := \{1, 2, ..., F\}$. Every application in the system uses a combination of functions from this set to perform its operations. Hence, each function can be associated with one or multiple applications, leading to shared or dedicated resource usage patterns. There are M distinct applications in the system. Each application $m \in \mathcal{M} := \{1, \ldots, M\}$ is modeled as a directed acyclic graph $G_m = (V_m, E_m)$, where V_m denotes the set of microservices or functions constituting the application, and E_m represents the directed edges indicating data dependencies between these functions.

In our model, we consider the fact that applications may potentially be malicious. Assigning resources to functions linked to malicious applications results in resource wastage as they do not contribute to the system's overall performance. We assume that for each application m, there exists a prior belief $\pi_m \in [0, 1]$, which indicates the probability of that application being malicious[2]. This prior knowledge influences the optimization problem, particularly in resource allocation decisions.

C. Performance Metrics

Each application has end-to-end latency and accuracy requirements, represented by $L_m \in \mathbb{R}^+$ and $A_m \in \mathbb{R}^+$ respectively. The latency of each function v within an application m, particularly when it is deployed on tier n, is influenced by the amount of allocated networking resources r^{net} and the number of times it is being used in the workflow. This usage of the function is characterized by the number of times it appears in the graphs of all applications and can be computed as $I_v = \sum_{m \in \mathcal{M}} \sum_{(u,v) \in E_m} w(u, v, m)$, where $w(u, v, m)$ is a weight function that returns the number of times the edge (u, v) exists in the graph of app m. Consequently, we define the latency function as $f_{v,n}(r^{\text{net}}, I_v) = a_n - b\frac{r^{\text{net}}}{I_v}$, where a_n represents the base latency in tier n, and b is a constant representing the rate of latency reduction per unit increase in network resources. Similarly, the accuracy of the function v is modeled by $g_{v,n}(r^{\text{com}}, I_v) = a'_n + b'\frac{r^{\text{com}}}{I_v}$, where a'_n signifies the base accuracy in tier n, and b' is a constant indicating how accuracy improves with additional

[2]Several risk assessment techniques are available in the literature to evaluate the prior belief [18] on applications being malicious.

computational resources. The system's overall performance is evaluated using a composite metric $\phi(L_m, A_m)$, where $\phi : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$ is a monotonically increasing function of both the latency L_m and accuracy A_m achieved by each application m.

III. Methodology & Analysis

The overarching objective is to judiciously allocate resources to functions and place them on suitable tiers. The goal is two-fold: minimizing the redundant resources used (factoring in maliciousness) and maximizing the system performance. This optimization is guided by a performance function ϕ that correlates the achieved thresholds for latency and accuracy.

A. Problem Formulation

The optimization problem can be mathematically formulated as follows:

$$\min_{\substack{x^n_{v,m}, r^{\text{net}}_{v,m}, \\ r^{\text{com}}_{v,m}, L_m, A_m}} \sum_{m\in\mathcal{M}} (\theta + \lambda\pi_m) \sum_{v\in V_m} (r^{\text{com}}_{v,m} + r^{\text{net}}_{v,m}) - \kappa \sum_{m\in\mathcal{M}} \phi(L_m, A_m), \tag{1}$$

$$\text{subject to} \quad \sum_{n\in\mathcal{N}} x^n_{v,m} = 1, \ \forall m \in \mathcal{M}, \forall v \in V_m, \tag{2}$$

$$\sum_{m\in\mathcal{M}} \sum_{v\in V_m} r^{\text{com}}_{v,m} x^n_{v,m} \le comp_n, \ \forall n \in \mathcal{N}, \tag{3}$$

$$\sum_{m\in\mathcal{M}} \sum_{v\in V_m} r^{\text{net}}_{v,m} x^n_{v,m} \le net_n, \ \forall n \in \mathcal{N}, \tag{4}$$

$$\sum_{v\in V_m} \sum_{n\in\mathcal{N}} f_{v,n}(r^{\text{net}}_{v,m}, I_v) x^n_{v,m} \le L_m, \ \forall m \in \mathcal{M}, \tag{5}$$

$$\sum_{v\in V_m} \sum_{n\in\mathcal{N}} g_{v,n}(r^{\text{com}}_{v,m}, I_v) x^n_{v,m} \ge A_m, \ \forall m \in \mathcal{M}. \tag{6}$$

The objective in (1) aims to minimize the effective resources used and maximize the allocation performance. The term $(\theta + \lambda\pi_m)$ serves as a weighting factor for the resources, increasing the cost of allocating resources to apps that have a higher prior belief of being malicious. The more likely an app is to be malicious (higher π_m), the more its resource use will count against the objective. The factor $\lambda \ge 0$ acts as a tuning parameter to adjust the influence of the belief about the app's maliciousness on the optimization. Several assumptions underpin our model as follows: (i) Each function can be deployed in one tier only. This is captured by the constraint (2), where binary decision variable $x^n_{v,m}$ indicates whether function v of application m is deployed on tier n; (ii) Resources allocated to a function do not exceed the tier's available capacities. This is captured by the constraints (3) and (4); (iii) Functions shared across applications experience increased latency based on their input load and reduced accuracy. This is ensured using constraints (5) and (6). The optimization problem described in (1)-(6) represents a mixed-integer nonlinear program (MINLP) due to the presence of both continuous and integer variables, as well as potentially nonlinear and objective functions. Given the complexity introduced by these factors, the problem can be shown to be NP-hard.

B. Observations and Insights

We now provide some key observations and insights based on the structure of the problem in the following lemmas and propositions. The impact of allocating compute and network resources to microservices is described by the following lemma:

Lemma 1. *Given the monotonically increasing nature of the functions f and g, for any function v, as we allocate more networking resources $r^{net}_{v,m}$, the latency will decrease and as we allocate more compute resources $r^{com}_{v,m}$, the accuracy will increase.*

The potential impact of deploying the microservices in different tiers on the latency and accuracy is described by the following lemma:

Lemma 2. *Considering monotonically increasing structure of base latency a_n, i.e., $a_1 < a_2 < \ldots < a_N$, and accuracy a'_n, i.e., $a'_1 < a'_2 < \ldots < a'_N$ in terms of the tier, deploying function v on a higher-tier (closer to the cloud) will have a higher base latency but potentially better accuracy due to more available resources. The reverse is true for lower tiers (closer to the edge).*

Proposition 1. *For any function v used by multiple applications, if function v is shared among multiple apps, it is better to allocate more resources to it than a function that is used by a single app, to improve resource efficiency.*

Proof. Given that multiple apps share function v, optimizing its performance ensures better aggregate performance across these apps. This improves the performance function ϕ in (1) with a smaller number of resources. □

The priority of allocating compute and network resources to microservices based on their roles in the applications is described by the following proposition:

Proposition 2. *If resources are limited, it is optimal to allocate resources first to functions that lie on the critical path of the applications' graphs. This is because these functions are the most sensitive in terms of the end-to-end latency and accuracy metrics.*

Proof. The critical path in the graph determines the end-to-end latency for the application. Therefore, optimizing functions on this path will have the highest impact on the overall application performance. □

C. Optimal Resource Orchestration Algorithm

To address the problem of resource allocation and performance optimization in this distributed multi-tier system, we propose a two-step iterative algorithm. The algorithm initially sets end-to-end latency L_m and accuracy A_m for each application m based on prior knowledge. It then iteratively optimizes resource allocation and function placement while adapting the performance metrics. Specifically, for fixed L_m and A_m, the algorithm solves an optimization problem to

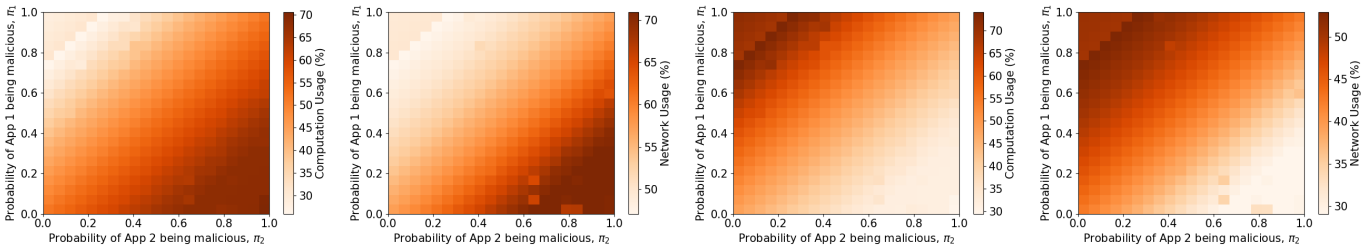


Fig. 2: Comparative analysis of computational and network usage percentages under varying probabilities of App 1 (π_1) and App 2 (π_2) being malicious. Each subplot represents (from left to right): a) Computational usage of App 1, b) Network usage of App 1, c) Computational usage of App 2, and d) Network usage of App 2.

Algorithm 1 Resource Allocation and Performance Optimization

Initialization:
Initialize L_m and A_m for all m
Set stopping criterion ϵ
while change in objective $> \epsilon$ **do**
  Step 1: Optimize Resource Allocation and Function Placement
  for each application m **do**
    Fix L_m and A_m
    Solve resource allocation problem
    Update $r^{\text{com}}_{v,m}$, $r^{\text{net}}_{v,m}$, and $x^n_{v,m}$
  end for
  Step 2: Update Performance Metrics
  for each application m **do**
    Determine new L_m and A_m
    Update L_m and A_m in the optimization problem
  end for
  Step 3: Update Objective Function
  Compute new objective value
  Check for convergence
end while

determine the best resource allocation $r^{\text{com}}_{v,m}$ and $r^{\text{net}}_{v,m}$, as well as the placement $x^n_{v,m}$ for each function. Subsequently, it updates L_m and A_m based on the newly achieved performance. Convergence is checked based on the change in the objective function. To solve the optimization problem, we can employ Lagrangian relaxation to decouple the constraints, thereby making the problem more tractable. The Lagrangian multipliers can be updated using the subgradient method. A pseudocode of the solution approach is provided in Algorithm 1.

D. Adaptive Re-orchestration Algorithm

Once, the malicious apps have been detected by the resource orchestrator, there is a need to re-allocate the resources used by malicious apps to existing functions in order to improve their performance and improve resource efficiency. Note that since functions have already been deployed, the goal of the re-orchestrarion process is to optimally re-arrange resources within each tier. Denote by $\mathcal{D}$ the set of detected malicious applications by the system operator, and $\{x^{n*}_{v,m}\}_{n\in\mathcal{N},v\in V_m,m\in\mathcal{M}\setminus\mathcal{D}}$ the solution to tier allocation of each function associated with applications obtained from the original optimization problem.

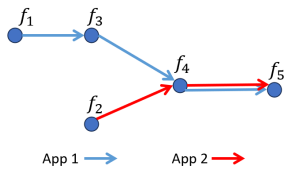


Fig. 3: Example workflows of applications using microservices f_1 through f_5. Both applications 1 and 2 share the resources allocated to microservices f_4 and f_5.

The optimal resource re-orchestration problem can be formulated as follows:

$$\min_{\substack{r^{\text{net}}_{v,m},r^{\text{com}}_{v,m},\\ L_m,A_m}} \sum_{\substack{m\in M\setminus D,\\ v\in V_m}} (r^{\text{com}}_{v,m}+r^{\text{net}}_{v,m}) - \kappa \sum_{m\in\mathcal{M}\setminus\mathcal{D}} \phi(L_m,A_m)$$

$$\text{subject to} \sum_{m\in\mathcal{M}\setminus\mathcal{D}} \sum_{v\in V_m} r^c_{v,m} x^{n*}_{v,m} \le comp_n,\ \forall n\in\mathcal{N},$$

$$\sum_{m\in\mathcal{M}\setminus\mathcal{D}} \sum_{v\in V_m} r^n_{v,m} x^{n*}_{v,m} \le net_n,\ \forall n\in\mathcal{N},$$

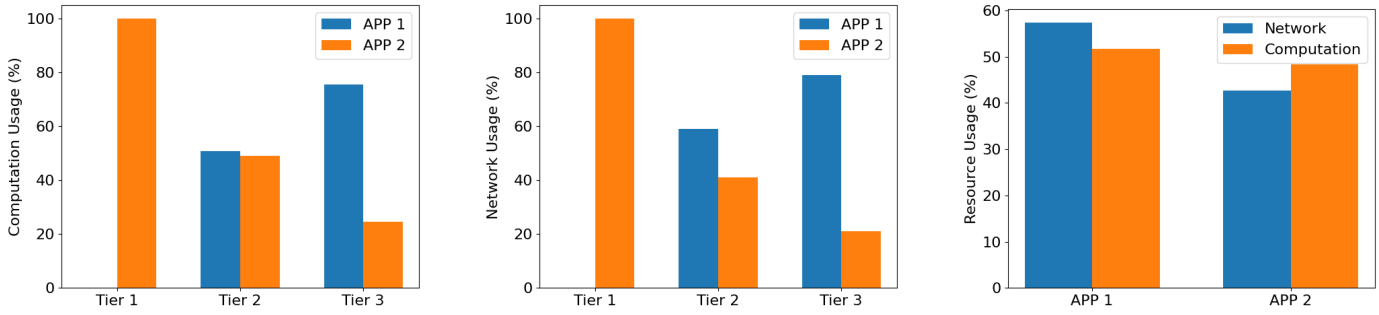
Constraints (5) and (6).

This problem, in general, constitutes a convex program and if ϕ is a linear function, it simplifies into a linear programming problem. Hence, this can be solved with a lower complexity than the original problem using standard convex optimization solvers.

IV. Performance Evaluation

A. Simulation Setup

Our simulation, designed to assess the effectiveness of the proposed algorithm, utilizes a three-tier resource orchestration system. We examine two different applications, each comprising five distinct functions. Fig. 3 illustrates the workflows of the two applications with graph G_1 shown in blue and graph G_2 shown in red. Functions f_4 and f_5 are shared between the two applications, creating a coupling relationship between the apps. The performance function ϕ is defined as $\phi(L,A) = -\alpha L + \beta A + \gamma$, where α, β, and γ are fixed constants. This definition implies that performance is inversely related to latency and directly related to accuracy. The specific values of all the simulation parameters are detailed in Table I.



(a) (b) (c)

Fig. 4: Analysis of resource utilization across three tiers for App 1 and App 2. Subplots illustrate: a) Compute resource usage by tier, b) Network resource usage by tier, and c) Aggregate resource utilization comparison between App 1 and App 2.

Parameters	Value
Total number of tiers, N	3
Number of functions, F	5
Number of applications, M	2
Capacity of compute resources, $comp^n$	7.65, 12.75, 17.00
Capacity of network resources, net^n	11.50, 21.85, 28.50
Weighting factor for performance function, κ	3e3
Penalty factor for malicious apps, λ	2e-5
Weighting factor for all apps, θ	5
Performance function constants (α, β, γ)	495, 499, 10

TABLE I: Simulation Parameters.

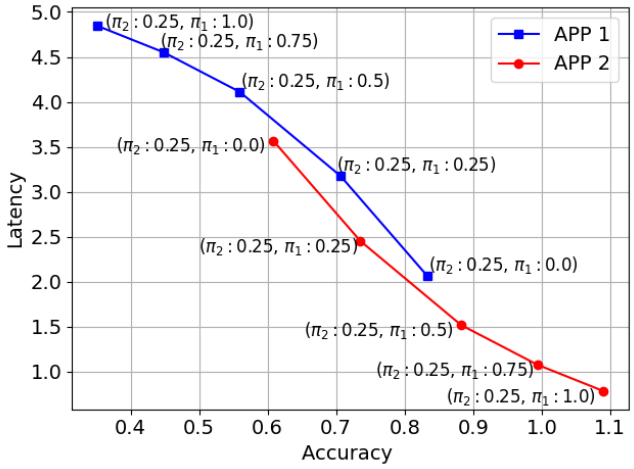
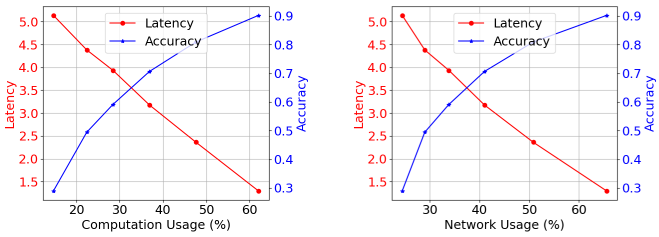


Fig. 5: Trade-off between latency and accuracy for App 1 and App 2 across varying belief of being malicious (π_2, π_1). Each data point represents a specific configuration of (π_2, π_1) values.

B. Experiment Results

To solve the MINLP problem of the resource orchestrator, we employed the Mixed-Integer Nonlinear Decomposition Toolbox in Python (MindtPy), which breaks down the non-convex optimization problem into a sequence of linear and continuous nonlinear optimization subproblems. In this subsection, we provide a sensitivity analysis and performance evaluation of the results based on the simulation scenario considered.

In Fig. 2, we examine the impact of the belief factor regarding the malicious nature of applications on resource allocation between computational and network resources. The



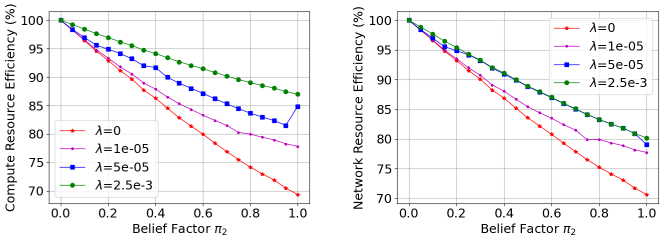
(a) (b)

Fig. 6: Trade-off analysis between system latency (in seconds) and application accuracy as a function of (a) computation usage (%) and (b) network usage (%).

results indicate a balance in resource allocation between the two applications when their belief factors π_1 and π_2 are equal. In contrast, a disparity in these factors shifts the resource allocation favoring the application perceived as less likely to be malicious.

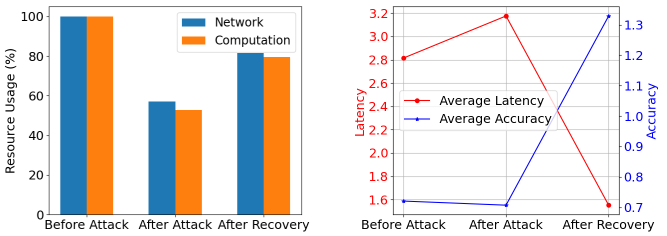
The relationship between the belief factor and application performance, in terms of latency and accuracy, is illustrated in Fig. 5. The findings highlight that enhanced resource allocation improves application performance, but this improvement for one application tends to compromise the other, showcasing the inherent trade-offs in orchestrating resources for multiple applications. Fig. 4 presents the resource distribution scenario when both applications are deemed non-malicious $\pi_1 = \pi_2 = 0$. In this case, computational and networking resources are nearly equally distributed between the two applications, achieving close to 100% utilization in each tier, thereby validating the direct correlation between resource allocation and application performance.

Fig. 6 shows the impact of resource usage on the application performance. The increment of both type of resource usage will generally improve the application performance albeit at a different rate depending on the latency and accuracy functions. Fig. 7 presents the average compute and network resource efficiency results for 50,000 realizations of simulated beliefs over malicious applications. We can observe that the resource efficiency generally decreases as the belief factor over the apps increases, affirming its role as a critical parameter for



(a) (b)

Fig. 7: Variation of resource efficiency with respect to belief factor for different lambda values: (a) Computation resource efficiency (%) and (b) Network resource efficiency (%).



(a) (b)

Fig. 8: Resilience of resource orchestration after malicious apps have been detected.

robust and efficient resource orchestration. Fig.8a illustrates the resource utilization patterns during and after a simulated malicious attack, under the condition that $\pi_1 = \pi_2 = 0$, indicating no initial suspicion of malicious activity. Post the attack, and subsequent to the detection and elimination of the malicious application, a noticeable decline in resource usage for the remaining application is observed. However, following the implementation of resource re-orchestration, there is a notable increase in resource allocation to the surviving application as shown in Fig.8b. This adjustment not only restores but potentially enhances both the performance and resource efficiency of the system.

V. Conclusion & Future Work

In this paper, we addressed the problem of optimizing the resource allocation and function placement within the multi-tiered, heterogeneous infrastructure of 5G networks for microservices based applications. Our approach considers end-to-end performance metrics (latency and accuracy), resource constraints at multiple tiers, and the potential for malicious applications. The objective is multi-faceted, aiming to minimize the effective use of resources while maximizing the overall system performance. We introduced coupling functions to capture the inter-dependencies between networking and compute resources, allowing a more holistic view of resource orchestration. The optimization problem is modeled as a MINLP and we proposed an iterative solution algorithm to solve it. Furthermore, we have also presented a re-orchestration algorithm that optimally rearranges resources once malicious apps have been detected and removed from the system. Results from simulations have revealed notable improvements in system performance, in terms of latency, accuracy, and overall resource utilization. These improvements were particularly pronounced in scenarios involving high risk of malicious applications, underscoring the effectiveness of our risk-aware resource allocation strategy. Future work may focus on developing efficient heuristics or approximation algorithms to solve this problem at scale.

References

[1] A. Ksentini and P. A. Frangoudis, "Toward slicing-enabled multi-access edge computing in 5G," *IEEE Network*, vol. 34, no. 2, pp. 99–105, 2020.

[2] J. Han, Y. Hong, and J. Kim, "Refining microservices placement employing workload profiling over multiple Kubernetes clusters," *IEEE Access*, vol. 8, pp. 192 543–192 556, 2020.

[3] Q. Luo, S. Hu, C. Li, G. Li, and W. Shi, "Resource scheduling in edge computing: A survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2131–2165, 2021.

[4] S. Agarwal, F. Malandrino, C.-F. Chiasserini, and S. De, "Joint VNF placement and CPU allocation in 5G," in *IEEE Conference on Computer Communications (INFOCOM 2018)*, 2018, pp. 1943–1951.

[5] Y. Wang, N. Li, P. Yu, W. Li, X. Qiu, S. Wang, and M. Cheriet, "Intelligent and collaborative orchestration of network slices," *IEEE Transactions on Services Computing*, vol. 16, no. 2, pp. 1239–1253, 2023.

[6] D. Sattar and A. Matrawy, "Optimal slice allocation in 5G core networks," *IEEE Networking Letters*, vol. 1, no. 2, pp. 48–51, 2019.

[7] J. Tang, J. Nie, Z. Xiong, J. Zhao, Y. Zhang, and D. Niyato, "Slicing-based reliable resource orchestration for secure software-defined edge-cloud computing systems," *IEEE Internet of Things Journal*, vol. 9, no. 4, pp. 2637–2648, 2022.

[8] Q. Liu, T. Han, and N. Ansari, "Learning-assisted secure end-to-end network slicing for cyber-physical systems," *IEEE Network*, vol. 34, no. 3, pp. 37–43, 2020.

[9] S. Kalafatidis and L. Mamatas, "Microservices-adaptive software-defined load balancing for 5G and beyond ecosystems," *IEEE Network*, vol. 36, no. 6, pp. 46–53, 2022.

[10] M. Zhu, X. Duan, H. Tu, Y. Wang, G. Zhou, X. Jin, and L. Zhao, "Microservice-based management and orchestration of 5G core network," in *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2022)*, 2022, pp. 609–615.

[11] X. Cheng, Y. Wu, G. Min, A. Y. Zomaya, and X. Fang, "Safeguard network slicing in 5G: A learning augmented optimization approach," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 7, pp. 1600–1613, 2020.

[12] B. Cao, Z. Sun, J. Zhang, and Y. Gu, "Resource allocation in 5G IoV architecture based on SDN and Fog-Cloud computing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 6, pp. 3832–3840, 2021.

[13] A. Gholami, K. Rao, W.-P. Hsiung, O. Po, M. Sankaradas, J. S. Baras, and S. Chakradhar, "Application-specific, dynamic reservation of 5G compute and network resources by using reinforcement learning," in *Proc. ACM SIGCOMM Workshop on Network-Application Integration*, New York, NY, USA, 2022.

[14] K. Ye, H. Shen, Y. Wang, and C.-Z. Xu, "Multi-tier workload consolidations in the cloud: Profiling, modeling and optimization," *IEEE Transactions on Cloud Computing*, vol. 10, no. 2, pp. 899–912, 2022.

[15] A. Ksentini and P. A. Frangoudis, "Toward slicing-enabled multi-access edge computing in 5G," *IEEE Network*, vol. 34, no. 2, pp. 99–105, 2020.

[16] N. Salhab, R. Rahim, R. Langar, and R. Boutaba, "Machine learning based resource orchestration for 5G network slices," in *IEEE Global Communications Conference (GLOBECOM 2019)*, 2019, pp. 1–6.

[17] A. Gholami, K. Rao, W.-P. Hsiung, O. Po, M. Sankaradas, and S. Chakradhar, "ROMA: Resource orchestration for microservices-based 5G applications," in *IEEE/IFIP Network Operations and Management Symposium (NOMS 2022)*, 2022.

[18] D. Yu, Y. Jin, Y. Zhang, and X. Zheng, "A survey on security issues in services communication of microservices-enabled fog applications," *Concurrency and Computation: Practice and Experience*, vol. 31, no. 22, p. e4436, 2019, e4436 cpe.4436. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.4436