Orchestrating Networked Robotic Applications

Khasa Gillani*,†, Jorge Martín-Pérez‡, Milan Groshev‡, Antonio de la Oliva‡, Carlos J. Bernardos‡, Robert Gazda‡

Abstract—In this letter, we formulate the orchestration problem for networked robotic applications considering contextual information. Any solution to the proposed formulation provides adequate routing updates, migration and radio handover decisions as the robot moves. We prove the NP-hard nature of the problem, and solve it for a remote driving robotic application with or without some contextual information, as in state-of-the-art. Results show that without contextual information it is impossible to meet the latency requirements of a remote driving robotic application.

Index Terms—robotic applications, orchestration, optimization

I. INTRODUCTION

NETWORKED robotic applications are being adopted to enhance operational automation and performance in some uses cases, e.g., assembly robots in Industry 4.0, or remotely controlled robots. However, in such use cases it is difficult to meet strict latency requirements [1] in the order of 1 ms, for the little computational resources of the robot make it difficult to compute some tasks on time. To overcome such limitation, recent works [2], [3] propose to split the networked robotics functionality into Virtual Functions (VFs), and offload them to servers with more computational resources.

Existing works as [2]–[4] solve an optimization problem to decide which servers should host the VFs of robotic applications, and how to steer traffic among them. We refer to such optimization problem as the orchestration problem. The main drawback of the literature is that it misses to consider some contextual information as network latency, or radio signal strength. As a result, the robotic application may not satisfy latency requirements if the traffic traverses congested links, or the robot attaches to Points of Access (PoA) with poor signal strength.

To that aim, this letter formulates an orchestration problem that (i) considers the contextual information from the network substrate and the robot-PoA communication; and (ii) imposes latency and radio constraints for the robotic application. Thanks to the contextual information, any solution of our problem formulation results into routing updates, VF migration and handover decisions that fulfill the latency requirements of latency-sensitive robotic applications.

II. ORCHESTRATION PROBLEM FORMULATION

In this section we formulate the orchestration problem for networked robotics. The goal is to consider contextual information to minimize the resource usage and take the adequate deployment, traffic routing and radio attachment decisions – see Fig. 1. Sections II-A to II-D specify the constraints imposed in the proposed formulation, and Section II-E proves its NP-hard complexity.

A. Computational constraints

The orchestration problem must ensure that the robotic application Virtual Functions (VFs) do not exhaust the computational resources \( C(n) \) of the hardware computing nodes \( n \in V(G) \) (i.e., vertices \( V(\cdot) \) of the hardware graph \( G \)):

\[
\sum_{v \in a(n)} C(v) \leq C(n), \quad \forall n \in V(G)
\]

That is, the computational requirements of all VFs \( v \) assigned to a computing node \( a(n) = \{v_1, v_2, \ldots\} \) must be lower than its available computational resources \( C(n) \). On top, it must ensure that all VFs \( V(S_i) = \{v_1, v_2, \ldots\} \) of a robotic application \( S_i \) are deployed at some computing node \( n \):

\[
\sum_{n \in V(G)} P(v, n) \geq 1, \quad \forall S_i \in S, v \in V(S_i)
\]

with \( P(v, n) = 1 \) if VF \( v \) is placed/deployed at the computing node \( v \in a(n) \), and 0 otherwise. Note that we do not prevent a VF to be “replicated”, or deployed in multiple nodes, since load-balancing may be required by some robotic applications coordinating multiple robots.
B. Network constraints

Once all VFs are deployed at computing nodes in the hardware graph $V(G)$, it is necessary to steer the traffic of the robotic application. Being aware of the bandwidth consumption $\lambda(n_1, n_2)$ over the hardware links $E(G)$ (i.e., edges $E(\cdot)$ of the hardware graph $G$), the orchestration problem must ensure that the link capacities are not exhausted:

$$\sum_{(v_1, v_2) \in E(n_1, n_2)} \lambda(v_1, v_2) \leq (1 - \delta(n_1, n_2)) \lambda(n_1, n_2), \forall (n_1, n_2) \in E(G)$$

with $\delta(n_1, n_2) \in [0, 1]$ the artificial packet drop rate that is present or injected at the link $(n_1, n_2) \in E(G)$. Specifically, all virtual links (VL) of an application $S_i \in S$ should not use all the link bandwidth $\lambda(n_1, n_2)$. In (3), $a(n_1, n_2) = \{(v_1, v_2), (v_2, v_3), \ldots\}$ denotes the VLs assigned/traversing the link $(n_1, n_2)$.

Also, inline with (3), orchestration problem has to process the traffic sent through the robotic application VLs $E(S_i)$:

$$\sum_{(v_1, v_2, n_1, n_2) \in E(G)} P(v_1, v_2, n_1, n_2) \geq 1, \forall S_i \in S, (v_1, v_2) \in E(S_i)$$

with $P(v_1, v_2, n_1, n_2) = 1$ if the VL $(v_1, v_2)$ is assigned to a link $(v_1, v_2) \in a(n_1, n_2)$, and $P(v_1, v_2, n_1, n_2) = 0$ otherwise. Hence, constraint (4) ensures that every VL of a robotic application $E(S_i)$ traverses at least one physical link $(n_1, n_2)$.

As the orchestration steers the VLs across the links, it must ensure that every switch $r$, and PoA $R_i$ ingresses and egresses the same amount of traffic, i.e., the flow constraint (5) must hold:

$$\sum_{(v_1, v_2) \in a(n_1, n_2)} \lambda(v_1, v_2) = \sum_{n \in \{v_1\} \cup \{R_i\}} \lambda(n_1, n_2), \forall (n_1, n_2) \in E(G)$$

In other words, all VLs $(v_1, v_2)$ entering a switch/Point of Access (PoA) $n$ from a prior hardware node $n_1$ - left hand side of (5) - should exit the switch/POA $n$ traversing another hardware node $n_2$ - right hand side of (5).

It is also important that every server $v_2 \in \{S_i\}$ processing VF $v_2$ should receive the corresponding VL traffic:

$$\sum_{(n_1, n_2) \in E(G)} P(v_1, v_2, n_1, n_2) = P(v_2, v_2) \forall (v_1, v_2) \in V(S_i)$$

otherwise, a solution of the orchestration problem may mistakenly steer the traffic to a server without VF $v_2$ running.

C. Delay constraints

The orchestration problem has to use contextual information to meet the delay constraints of robotic applications as remote driving. Namely, it has to consider both the propagation and processing delay of the sensors’ data sent by a robot that is driven remotely.

The network delay experienced by the robotic application $S_i$ is the sum of the delay of links $d(n_1, n_2)$ traversed by the VLs $(v_1, v_2)$, and an artificial queuing delay $\psi(n_1, n_2)$ that may be present/injected in network to introduce latency heterogeneity:

$$d_{net}(S_i) = \sum_{(v_1, v_2) \in E(S_i)} \sum_{(n_1, n_2) \in E(G)} d(n_1, n_2) + \psi(n_1, n_2), \forall S_i \in S$$

To compute the processing delay we resort to the M/G/1-PS expression for the average delay [6], for it is a common practice in the existing literature [7–9]. Therefore, we obtain the processing delay of a VF $v$ as:

$$d_{pro}(v) = \sum_{(v_1, v) \in E(S_i)} \frac{1}{C(v)} \mu - \lambda(v_1, v), \forall S_i \in S, v \in V(S_i)$$

where $\mu$ is the processing rate of a CPU within the hardware graph. That is, we have an M/G/1-PS system with an aggregate processing rate $C(v)$ and an arrival rate $\lambda(v_1, v)$, i.e., the incoming traffic to the VF $v$.

Hence, a solution of the orchestration problem must ensure that the network and processing delay remain below the requirement of the robotic application $D(S_i)$:

$$d_{net}(S_i) + \sum_{v \in V(S_i)} d_{pro}(v) \leq D(S_i), \forall S_i \in S$$

D. Radio constraints

Since robotic applications leverage wireless technologies to connect with VFs running in the network substrate, it is of paramount importance that the orchestration problem formulation prevents robots from using radio links that cannot meet the robotic application requirements. In other words, it is mandatory to consider radio constraints.

The first constraint to impose is that a VL $(v_1, v_2)$ cannot traverse the link connecting the robot with the PoA $(r_i, R_i)$ unless the robot wireless interface is attached to the PoA:

$$P(v_1, v_2, r_i, R_i) \leq \phi(r_i, R_i), \forall (v_1, v_2), r_i, R_i$$

with $\phi(r_i, R_i) = 1$ if the orchestration tells the robot $r_i$ to attach to the PoA $R_i$, and zero otherwise.

Note that any solution of the orchestration problem has to ensure that the robot $r_i$ network interface is attached to one PoA $R_i$ to have connectivity:

$$\sum_{R_i} \phi(r_i, R_i) = 1, \forall r_i \in \{r_i\}, i$$

otherwise, any robotic application $S_i$ will not have the chance of processing its traffic at a remote server running a VF $v \in V(S_i)$.

Since the wireless connectivity suffers from background noise $N$ and heavily depends on the signal strength $\sigma_{R_i}(r_i)$, it is necessary to account for the effective bandwidth capacity over the wireless link from the robot to the PoA $(r_i, R_i)$. Inline with recent works as [10], we model the wireless transmission capacity as:

$$T(r_i, R_i) = (1 - \delta(r_i, R_i)) \lambda(r_i, R_i) \log_2 \left(1 + \frac{\sigma_{R_i}(r_i)}{N}\right)$$
with \((1 - \delta(r_i, R_i))\lambda(r_i, R_i)\) being the perfect conditions bandwidth over the wireless link \((r_i, R_i)\), considering the packet loss \(\delta(r_i, R_i)\); and the log2\((1 + \sigma_R(r_i)/N)\) being the attenuation term under the presence of noise given a certain signal strength. Note that we assume \(N\) is additive Gaussian white noise.

Knowing the bandwidth constraint discussed in (5), if one accounts for the wireless transmission capacity \(T(r_i, R_i)\) of the link in between the robot \(r_i\) and the PoA \(R_i\), the orchestration problem must impose:

\[
\sum_{(v_1, v_2) \in a(r_i, R_i)} \lambda(v_1, v_2) \leq T(r_i, R_i), \quad \forall (v_1, v_2) \in a(r_i, R_i) \tag{13}
\]

so all VLs \((v_1, v_2)\) traversing the robot-to-PoA wireless connection do not exceed the transmission capacity.

### E. Problem statement and complexity

Thanks to the contextual information, we can impose the constraints presented in the prior sections for the deployment of a networked robotic application. The goal is to minimize the used resources, so the substrate network is shared by multiple robotic applications. In the following we present the problem formulation proposed by this letter.

**Problem 1** (Orchestration of networked robotic applications). Given the computational constraints (1)–(2), network (5)–(6), delay (9), and radio constraints (10)–(11), of robotic applications \(S_i\); minimize the resource usage by taking the adequate VF deployment \(P(v, n)\), routing \(P(v_1, v_2, n_1, n_2)\), and attachment \(\phi(r_i, R_i)\) decisions.

\[
\begin{align*}
\min_{P(\cdot), \phi(\cdot)} & \sum_{n \in V(G)} |a(n)| + \sum_{(n_1, n_2) \in E(G)} |a(n_1, n_2)| \\
\text{s.t. : } & (1) - (6), (9) - (11), (13)
\end{align*}
\]

However, finding the optimal solution of Problem 1 is not straightforward. Inline with the complexity of existing orchestration problems (5)–(11), in the following Lemma 4 we prove that it is NP-hard.

**Lemma 4.** Solving Problem 1 is NP-hard.

**Proof.** We proof that our proposed model is NP-hard showing that an instance of Problem 1 is equivalent to the bin-packing problem [12]. Let us consider a set of “ideal” robotic applications \(S_i\) without any delay \(D(S_i) = \infty\), \(\forall S_i \in S\) and bandwidth \(\lambda(v_1, v_2) = 0\) requirements. On top, these “ideal” robotic applications consist of just two VFs \(v_{r_1}, v_{r_2}\) and \(V(S_i), \forall S_i \in S\), a single VL \((v_{r_1}, v_{r_2})\) and the former VF has to run in the robot \(P(v_{r_1}, r_1) = 1\), \(\forall S_i \in S\) – see (Fig. 2). Hence constraints (2) and (4) are strict equalities.

Now let’s assume that all these “ideal” robotic applications have to be deployed in a hardware graph consisting of one robot \(r_1\) with infinite computational capacity \(C(r_1) = \infty\), a single PoA \(R_1\) that covers all the geographical area with adequate signal strength towards the robot, so \(\sigma_R(r_1) > N\) holds, and an infinite number of servers \(\{s_i\}_N^\infty\) with finite capacity \(C(s_i) < K \in N^+\). All the servers have at one hop distance the PoA \(R_1\), as shown in Fig. 2.

Hence, in the resulting scenario – depicted in Fig. 2 – the orchestration process just has to decide where to deploy the second VF \(P(v_{r_2}, s_1)\), and an infinite pool of servers \(\{s_i\}_N^\infty\). Deciding in which server we place/deploy the second VF \(P(v_{r_2}, s_1)\) is the bin-packing problem, thus, it is NP-hard.

### III. Model validation

To validate the formulation, we solve our orchestration Problem 1 to take deployment, routing, and attachment decisions of a remote driving robotic application \(S_0\) with two VFs \(V(S_0) = \{v_1, v_2\}\). The former VF \(v_1\) runs the robot drivers, and the latter VF \(v_2\) contains the remote driving logic. The latency requirement of the remote driving robotic application is \(D(S_0) = 15\ms\). The experiment setup consists of six PoAs \(R_1, \ldots, R_6\) in a hallway (see Fig. 1) with different latencies towards Near/Far Edge and Cloud servers (see TABLE). As the robot \(r_0\) moves; the signal strength \(\sigma_R(R_i)\) towards the PoAs change, and a commodity solver (implemented using Gurobi v9.5.0) finds solutions of Problem 1 to update the corresponding placement \(P(v_1, n)\), attachment \(\phi(r_0, R_i)\) and traffic routing \(P(v_1, v_2, n_1, n_2)\) decisions of the robot drivers VF \(v_1\) and remote driving VF \(v_2\).
We first validate if the decisions taken by the solver (Gurobi) meet the 15 ms latency requirement by iteratively solving Problem 1 while updating the contextual information. In Fig. 3 (a) we see that the 15 ms latency requirement is met because by the time the robot attaches to PoAs $R_2, \ldots, R_5$ with high delay to the Cloud, the orchestration migrates the remote driving VF to the Far Edge (orange) – with smaller delay to the PoAs.

We then check in Fig. 3 (b) what would happen if we ignore the radio signal information, as done in [4]. The result is that the orchestration solution instructs PoA attachments regardless of their signal strength $\sigma_{R}(r)$. For instance, it instructs attachments to PoA $R_5$ during the first/last 150 sec., with such PoA not having enough signal strength $\sigma_{R}(r) \simeq 0$ to satisfy the wireless transmission capacity constraint (12). Consequently, the robot loses connectivity.

Finally, we check in Fig. 3 (c) what happens if we ignore the links’ delay $d(n_1, n_2)$ information, as done in [2]. We see that the solution does not migrate the remote driving VF out of the Cloud when the robot is attached to PoAs with high delay towards the Cloud. This is because it cannot check whether the latency constraint (9) is met. Thus, during the interval 100-300 sec., robot is attach to PoAs $R_2, \ldots, R_6$; for it does not know that their latencies towards the Cloud exceed the 15 ms (see TABLE. 1).

**IV. CONCLUSION**

In this letter we formulate the orchestration problem for networked robotic applications to take routing, robot-PoA attachment, and VF deployment & migration decisions. Any solution of our problem meets the latency requirements of networked robotic applications using contextual information from the PoAs and network substrate. We validate the problem formulation by solving it in a latency-sensitive remote driving robotic application that runs its VFs in the Cloud/Edge as it travels through a corridor. Contrary to the state of the art, solutions of our problem formulation meet the latency requirement of the remote driving robotic application.

**TABLE I**

| PoA | Cloud | Far Edge | Near Edge |
|-----|-------|----------|-----------|
| $R_1$, $R_3$ | 9 ms | 4 ms | 3 ms |
| $R_2$, $R_4$ | 18 ms | 8 ms | 9 ms |
| $R_5$, $R_6$ | 27 ms | 12 ms | 9 ms |

**REFERENCES**

[1] 3GPP, “Service requirements for cyber-physical control applications in vertical domains,” 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 22.104, 21 2021.

[2] C. Delgado, L. Zanzi, X. Li, and X. Costa-Pérez, “OROS: Orchestrating ROS-driven Collaborative Connected Robots in Mission-Critical Operations,” arXiv preprint arXiv:2205.03256, 2022.

[3] W. Chen, Y. Yaguchi, K. Naruse, Y. Watanobe, and K. Nakamura, “QoS-aware robotic streaming workflow allocation in cloud robotics systems,” IEEE transactions on services computing, vol. 14, no. 2, pp. 544–558, 2018.

[4] B. Németh, N. Molner, J. Martín-Pérez, C. J. Bernardos, A. de la Oliva, and B. Sonkoly, “Delay and Reliability-Constrained VNF Placement on Mobile and Volatile 5G Infrastructure,” IEEE Transactions on Mobile Computing, vol. 21, no. 9, pp. 3150–3162, 2022.

[5] H. Feng, J. Llorca, A. M. Tulino, D. Raz, and A. F. Molisch, “Approximation algorithms for the NFV service distribution problem, year=2017,” in IEEE INFOCOM 2017 - IEEE Conference on Computer Communications, pp. 1–9.

[6] Leonard Kleinrock, Queueing Systems, Volume I. Wiley, 1975.

[7] R. Cohen, L. Lewin-Eytan, J. S. Naor, and D. Raz, “Near optimal placement of virtual network functions,” in IEEE INFOCOM, 2015, pp. 1346–1354.

[8] F. B. Jemaa, G. Pujolle, and M. Pariente, “QoS-aware VNF placement optimization in edge-central carrier cloud architecture,” in 2016 IEEE Global Communications Conference (GLOBECOM), IEEE, 2016, pp. 1–7.

[9] D. B. Olijira, J.-K. Grinnemo, J. Täheri, and A. Brunstrom, “A model for QoS-aware VNF placement and provisioning,” in 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), IEEE, 2017, pp. 1–7.

[10] L. Bonann, S. D’Oro, L. Bertizzolo, E. Demirors, Z. Guan, S. Basagni, and T. Melodia, “CellOS: Zero-touch Softwarized Open Cellular Networks,” Computer Networks, vol. 180, p. 107380, 2020.

[11] Y. Song, B. Ji, G. R. Gupta, X. Du, and L. Ye, “Provably efficient algorithms for joint placement and allocation of virtual network functions,” in IEEE INFOCOM, 2017.

[12] Garey, Michael R. and Johnson, David S., Computers and Intractability: A Guide to the Theory of NP-Completeness. USA: W. H. Freeman & Co., 1990.