Architectural Exploration and Design for Ultra-Reliable Low-Latency Indoor Robotics Systems

Ayub Shah and Roberto Passerone
Department of Information Engineering and Computer Science
University of Trento, Trento, Italy 38123
Email: first.last@unitn.it

***Abstract*—Modern day communication systems are evolving to support computation-intensive and communication-sensitive applications that impose diverse quality of service requirements on the network in terms of latency, reliability, and bandwidth. Applications such as autonomous vehicles, industrial automation, and remote surgery will require ultra-reliable and low-latency communication, paving the way towards resourceful servers closer to user, i.e., multi-access edge computing. However, the stringent requirements on both communication and computation make effective network control difficult. The high dynamicity of the involved processing and interaction patterns requires planning and deployment of architectures with optimal design and cross-optimization of computation and communication resources. In this work, we address the design problem with QoS guarantees and architectural exploration and optimization, to provide computing and communication resources, accounting for dynamic mobility, traffic, and application patterns in the context of edge servers.**

***Index Terms*—URLLC, MILP, Design Space Exploration, Edge Computing**

I. Introduction

The evolution of mobile communication in modern day goes beyond the limits of traditional communication systems by enabling URLLC (ultra-reliable low-latency communication) whose minimum latency is 1 ms [1], [2] and by making the network intelligent [3]. For the URLLC between a network service and its mobile user (e.g., between an autopilot service and an automated guided vehicle (AGV) and between the virtual-guide service of a museum and a museum visitor), the target latency is achieved by using MEC (multi-access edge computing) that allows network services to run very close to their mobile users. 5G and B5G presents a break with 4G by making the network intelligent, not to mention the tremendous increase in network density and heterogeneity [4], [5], network providers face a more complex network design task. This is particularly true when the network providers would like to minimize their cost while guaranteeing the service providers certain QoS (quality of service) levels, such as guaranteeing to a service the required computation and communication bandwidth, and a minimum latency [6]. To manage the network design complexity, network providers can benefit from design-space exploration (DSE) tools, such as ArchEx [7], which allow network designers to optimally satisfy a set of requirements (e.g., the various QoS levels required by different services) by choosing a set of components (e.g., Access Points, Servers, and others) from a network component library [8].

ArchEx is implemented as a MATLAB toolbox that works following the flow depicted in Figure 1. Starting from the top left of the figure, the implementation takes as input a set of files, including a problem description that expresses the directed graph, system properties, and objective function, and a library file that contains a list of components and their attributes. In the next step, the implementation translates the given input into YALMIP [9], which is a MATLAB toolbox to easily formulate a mixed-integer linear program (MILP) problem. To solve the MILP problem, YALMIP in turn uses one of the many MILP solvers that it supports, such as GLPK[1] and CPLEX[2]. In the last step, the implementation produces as output an optimized architecture that is displayed using MATLAB.

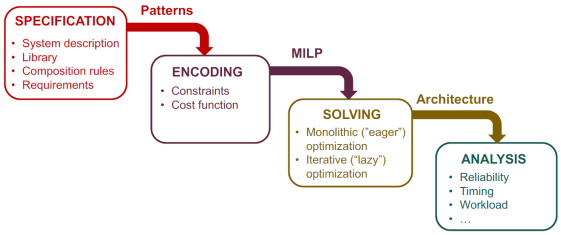


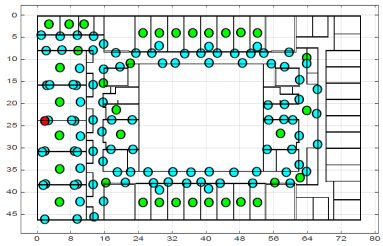
Fig. 1. The processing flow of ArchEx.

Designing a URLLC based indoor robotics network for a large factory that hosts robots, servers, APs, and services, a network designer first chooses a number of network components and places them on the model of the factory (e.g., a floor plan). Then, the designer runs the DSE tool to obtain the most inexpensive off-the-shelf products for every chosen network components while still satisfying the QoS levels required by the services. This, however, is not the only possibility because as demonstrated using ArchEx the designer can indeed obtain other design answers (e.g., which path between a robot ⇒ service are redundant) and ask other design questions [10], [11]. For instance, instead of manually placing the chosen components on the factory model,

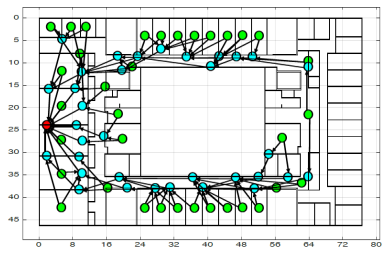
[1]https://www.gnu.org/software/glpk/
[2]https://www.ibm.com/analytics/cplex-optimizer

the designer can specify the desired received signal strength (RSS) and then let the tool place the components on the model automatically. To demonstrate the design effectiveness of ARCHEX, particularly to highlight the potential benefits of using ARCHEX in designing URLLC based networks, we adopted the methodology of a wireless sensor networks (WSN) system as an example and extended it further because WSN is not only the example most relevant to indoor robotics scenario but also an underpinning communication technology in both cyber-physical and internet-of-things systems.

Our starting point is the WSN architectural design in [10], which is concerned with the design of an indoor static network topology that considers a fixed physical placement of nodes and their fixed physical links in an indoor setting, and the selection of network components to be deployed. The architectural design is demonstrated on two different system: data collection and localization. The architectural design is expressed using three text files, i.e., a problem description and a library, and one SVG (scalable vector graphics) file which expresses an indoor floor plan. As illustrated in Figure 2(a), the SVG file expresses obstacles (e.g., walls), their dimensions, and the placement of network components (i.e., nodes). Given the three input files, ARCHEX goes through the process shown in Figure 1 and produces as output not only the network topology as illustrated in Figure 2(b) but also the selection of needed network devices that are all optimal with respect to the system objective.



(a) The floor plan input to ARCHEX. Source: Figure 1(a) of [10].



(b) The floor plan output from ARCHEX. Source: Figure 1(b) of [10].

Fig. 2. The WSN system for data collection.

Using ARCHEX, a system designer first expresses a system architecture in the form of a directed graph, which is a set of nodes and their interconnecting links, where the nodes and links have specific attributes and are chosen from a pre-defined component library. The system designer can then use a pre-defined pattern language to specify a set of system properties and an objective function before running ARCHEX to obtain an optimal system architecture. Considering that ARCHEX has shown its effectiveness in the design of several cyber-physical systems (CPS) [7], [10], our motivation is to target its static nature and overcome this limitation by designing a dynamic scenario where robots follow a trajectory and continuously change their positions at each discrete time step. Our goal is to exploit the scale-able, express-able, modular, and adoptable capabilities of ARCHEX and then demonstrate various interesting ways a designer can explore the design space of a URLLC based network that hosts a set of various services and maintains their QoS levels. Therefore, our main contributions in this paper are:

- Introduce a time dimension t in the problem formulation, to support the creation of dynamic scenarios where objects and connectivity can change during execution.
- Create a new set of specification patterns for QoS requirements, with variables supporting computation, communication, and latency requirements, respectively. These variables will be denoted as α, β and λ.
- Introduce a novel case study of a dynamic nature for an indoor robotic scenario and its MILP-based formulation.

To achieve this, we also update the area-map processing to take trajectories into account, and build the necessary connection data structures. Section II presents the system model and formulation, while Section III evaluates the designed MILP model effectiveness and results in various scenarios. Finally, Section IV outlines our conclusions and future work.

II. SYSTEM MODEL & MILP FORMULATION

To formalize the architecture, we adopt the baseline methodology of ARCHEX and extend the already implemented basis constraints by introducing a time dimension and implementing a new set of constraints as per our case study to get an optimal and feasible network architecture.

We employ a directed graph model $G = V, E$, where V is a set of components (nodes) $V_1, \ldots, V_{|V|}$, $|V|$ being the cardinality of V. E is a set of edges $e_{ij} \in E$ present in the network. Both nodes and edges in the graph are labeled with types, terminal variables, and attributes corresponding to those from the library L. The edges $e_{ij} \in E$ of G are binary variables presenting a connection between v_i and v_j, where $e_{ij} = 1$ (0) indicates the presence (absence) of connections between the source and destination nodes. Similarly, a node v_i is represented by a binary variable δ_i which evaluates to 1 if at least one incoming edge e_{ji} or outgoing edge $e_{ij} = 1$.

We denote with $M : V \to L$ the map that associates each *virtual* component of the template with a *real* component in the library. We represent this map with a binary variable $m_{ij} \in M$, which is $= 1$ if $v_i \in V$ is mapped to $l_j \in L$, else $= 0$. So, given a template $T = (V, E)$ and a library of components L, we use the optimization to find a topology E^* and a map M^* to satisfy a set of requirements (e.g., interconnection, routing, link-quality, and QoS), while minimizing a cost function. Next, we formulate the system requirements and applications constraints for, but not limited to, QoS, routing, and mapping constraints in terms of mixed-integer linear constraints.

A. *Path Constraints*

A network path π is defined by a sequence of distinct nodes $v_0, \ldots, v_n$. The edges $e_{ij} \in E$ are labeled with a binary variable y_{ij}^{π} which is 1 if the edge e_{ij} connects the pair of nodes (v_i, v_j) in the path π, else is 0. Similarly, a node $v \in V$ is labeled with a binary variable w^{π} and $w_i^{\pi} = 1, (0)$ if node v_i belongs to (not) path π. Equation (1) is a balance equation which is an integer linear constraint imposed to make sure that the values assigned to path and node variable (y^{π}, w^{π}) are consistent with the topology configuration.

$$c(y_t^{\pi})^T = z_t^{\pi} \quad t \in \mathbb{R} \tag{1}$$

where c is the incidence matrix of size $|V| \times |E|$ that is 1 (-1) if the edge leaves (enters) the node, and 0 otherwise. z^{π} is a column vector of size $|V|$ in which $z_{src}^{\pi} = 1$, $z_{dst}^{\pi} = -1$, and 0 otherwise. Next, a constraint defines the relation between e_{ij} and y_{ij}^{π}.

$$y_{ijt}^{\pi} \leq e_{ij} \quad \forall i, j \in \mathbb{N} : 1 \leq i, j \leq |V|, t \in \mathbb{R} \tag{2}$$

Constraint (2) defines that the edge e_{ij} is present only if it is used by a path at any time t.

B. *Mapping Constraints*

The mapping constraints are labeled with a binary variable m_{ij} which evaluates to 1 only if a virtual component from template T is associated to a real component from library L. The variables m_{ij}, called mapping variables, are grouped into a set M. To ensure the consistency of M, a set of linear constraints are imposed.

$$\bigvee_{i=1}^{|L_k|} |m_{ij}^k| = \bigvee_{i=1}^{|V|} (e_{ij} \vee e_{ji}) \quad \forall j \in \mathbb{N} : 1 \leq j \leq |P_k| \tag{3}$$

Constraint (3) imposes that each node v of type k that is instantiated must be mapped to one of the components in L_k, where L_k represents the type of the components in the library.

C. *QoS Constraints*

QoS constraints are divided in three different groups, i.e., computation, communication, and latency, represented as α, β, and λ, respectively. QoS constraints are linear integer constraints imposed to satisfy the maximum and minimum α, β capacity requirement of a specific service and to maintain a minimum λ for a service execution.

$$\alpha_{i,j}^{\min} \leq \alpha_{i,j,t} \leq \alpha_{i,j}^{\max} \tag{4}$$

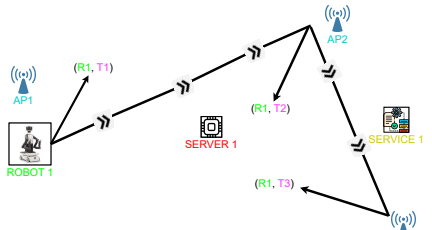
$$\beta_{i,j}^{\min} \leq \beta_{i,j,t} \leq \beta_{i,j}^{\max} \tag{5}$$

$$\lambda_{i,j,t}^{\min} \leq \lambda_{i,j}^{\max} \tag{6}$$

In the constraints (4) and (5) the two real variables, computation bandwidth $\alpha_{i,j,t}$ and the communication bandwidth $\beta_{i,j,t}$ given to a service at any time t must be decided optimally, while $\alpha_{i,j}^{\min}$ and $\beta_{i,j}^{\min}$ are the least bandwidth and $\alpha_{i,j}^{\max}$ and $\beta_{i,j}^{\max}$ are the maximum bandwidth. Finally, another important aspect of the model is the end-to-end latency of the path. Each route that a robot must follow has to experience an end-to-end latency. The constraint in (6) has to satisfy the upper-bound threshold latency $\lambda_{i,j}^{\max}$ beyond which the robot associated the service would fail in executing its real-time tasks.

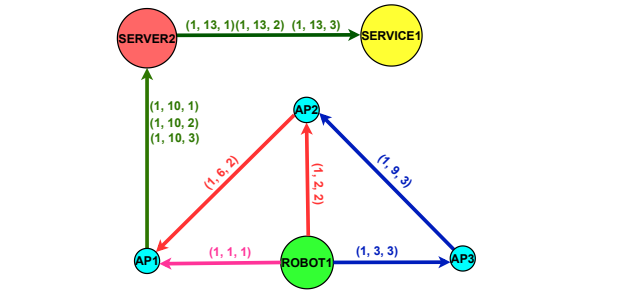
III. Evaluation

We initially evaluate the model for two scenarios and for each scenario we create a separate template and library text files and a SVG file. Our goal is to get an optimal solution and let the optimization tool decide to choose a feasible and unique path for the robot at every time step. The first scenario is shown in Figure 3, Figure 3(a) is the input SVG which represents an indoor building environment (for simplicity all unwanted obstacles are removed) and Figure 3(b) shows the final graph generated by ArchEx after the optimization runs successfully and the optimizer finds the optimal solution to the problem.



R1 (--------) represents the robot index in Template
T1,..., T3 (----------) represents the position of robot at discrete time steps

(a) The floor plan input to ArchEx.



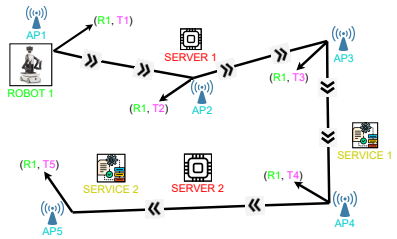
(b) The output from ArchEx.

Fig. 3. Scenario 1.

In the SVG, the AP nodes, being the medium of path between source and destination nodes, are placed in such a manner that at each time step when the robot changes its position it must receive a min RSS to have connectivity and find a path to the destination node, i.e., the service. We create all three input files having the following number of nodes: $Robot = 1, AP = 3, Server = 1$, and $Service = 1$, and also the robot trajectories at different coordinates showing its direction and position at every discrete time step.

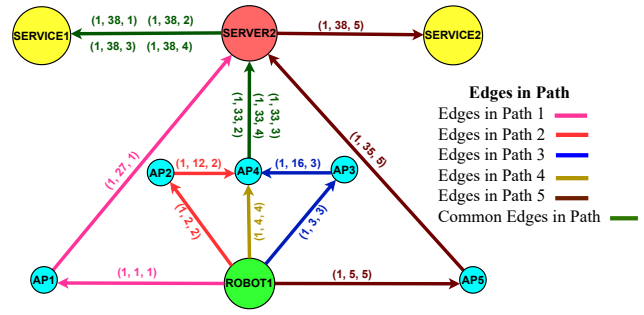
Similarly, the second scenario is shown in Figure 4. The scenario is created with robot having more trajectory points, $Robot = 1, AP = 5, Server = 2$, and $Service = 2$ as shown in Figure 4(a). After the successful execution of the problem

and once the optimal solution is found, we get Figure 4(b) as the output in which the paths and node selection and placement is performed by ARCHEX.



R1 (--------) represents the robot index in Template
T1,..., T5 (----------) represents the position of robot at discrete time steps

(a) The floor plan input to ARCHEX.



Path = (R, E, T)
First variable "R" represents the robot index in the Path
Second variable "E" represents the edge index in the Path
Third variable "T" represents the robot position at discrete trajectory points

(b) The output from ARCHEX.

Fig. 4. Scenario 2.

In Scenario 1, we first compute a connection_matrix and a threshold_matrix which are 3×3 matrices as shown in Table I and Table II, respectively. In scenario 2, we again compute both matrices which are 5×5 matrices as shown in Table III and Table IV, respectively. The connection_matrix is computed from a path loss (PL) function [12] that uses the euclidean distance and takes as input a distance_threshold (d) (computed from the channel model and geometry) and the two vectors of coordinate pairs (robot and AP). The threshold_matrix is computed by comparing the elements of the connection_matrix to a receive threshold value RX_threshold to see if the AP is in range. The threshold_matrix is then implemented with the path constraints as defined in Section II and lets the optimizer find a unique path between the robot and its associated service.

TABLE I
CONNECTION MATRIX1

Robot,Time	AP1	AP2	AP3
R1,T1	64.9339	101.1238	106.3163
R1,T2	102.3847	63.3013	102.9831
R1,T3	106.3328	99.3281	56.0388

TABLE II
THRESHOLD MATRIX1

AP1	AP2	AP3
1	0	0
0	1	0
0	0	1

IV. CONCLUSION AND FUTURE WORK

In this paper, we present a novel case study in which we re-formulated the MILP optimization problem, develop, and implement a dynamic scenario in ARCHEX. Different from previous case studies in ARCHEX, we designed a network

TABLE III
CONNECTION MATRIX2

Robot,Time	AP1	AP2	AP3	AP4	AP5
R1,T1	62.80	96.74	108.30	107.21	98.98
R1,T2	94.60	68.19	100.74	101.73	100.99
R1,T3	106.92	100.00	57.48	99.98	108.97
R1,T4	106.09	101.34	98.71	56.90	105.22
R1,T5	98.76	101.29	109.32	106.18	55.69

TABLE IV
THRESHOLD MATRIX2

AP1	AP2	AP3	AP4	AP5
1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

where a new group of patterns are developed and introduced a dynamic feature, i.e., the time dimension for a URLLC based indoor robotics scenario with the objective to minimize the overall network cost by satisfying topology, mapping, and application constraints. Considering the latency, reliability, and QoS requirements, the formulated MILP optimization model takes into account path redundancy where robots and their associated services are always connected with resourceful servers through different paths at every time step. Our aim for future work is to implement a complex scenario by increasing the number of robots, APs, servers, and services, evaluate the service migration, and perform simulation in-the-loop with the optimization to verify the network architecture.

REFERENCES

[1] S. Z. Asif, *5G Mobile Communications: Concepts and Technologies*. Boca Raton, FL: CRC Press, 2018.

[2] M. Agiwal, A. Roy, and N. Saxena, "Next generation 5g wireless networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 1617–1655, 2016.

[3] F. H. Fitzek, P. Seeling, T. Höschele, and B. Jacobfeuerborn, "On the need of computing in future communication networks," in *Computing in Communication Networks: From Theory to Practice*, F. H. Fitzek, F. Granelli, and P. Seeling, Eds. UK: Academic Press, 2020, pp. 3–45.

[4] Y. Teng, M. Liu, F. R. Yu, V. C. M. Leung, M. Song, and Y. Zhang, "Resource allocation for ultra-dense networks: A survey, some research issues and challenges," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2134–2168, 2018.

[5] N. Zhang, N. Cheng, A. T. Gamage, K. Zhang, J. W. Mark, and X. Shen, "Cloud assisted hetnets toward 5g wireless networks," *IEEE communications magazine*, vol. 53, no. 6, pp. 59–65, 2015.

[6] T. Prastowo, A. Shah, L. Palopoli, R. Passerone, and G. Piro, "Migration-aware optimized resource allocation in B5G edge networks," in *Proceedings of the 19th IEEE Consumer Communications & Networking Conference*, Las Vegas, NV, January 8–11, 2022.

[7] D. Kirov, P. Nuzzo, R. Passerone, and A. L. Sangiovanni-Vincentelli, "ArchEx: An extensible framework for the exploration of cyber-physical system architectures," in *Proceedings of the 54th Design Automation Conference*, Austin, TX, June 18–22, 2017.

[8] D. Kirov, P. Nuzzo, A. L. Sangiovanni-Vincentelli, and R. Passerone, "Efficient encodings for scalable exploration of cyber-physical system architectures," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2023.

[9] J. Lofberg, "Yalmip: A toolbox for modeling and optimization in matlab," in *2004 IEEE international conference on robotics and automation*. Washington, DC, USA: IEEE, 2004, pp. 284–289.

[10] D. Kirov, P. Nuzzo, R. Passerone, and A. L. Sangiovanni-Vincentelli, "Optimized selection of wireless network topologies and components via efficient pruning of feasible paths," in *Proceedings of the 55th Design Automation Conference*, San Francisco, CA, June 24–28, 2018.

[11] P. Nuzzo, N. Bajaj, M. Masin, D. Kirov, R. Passerone, and A. L. Sangiovanni-Vincentelli, "Optimized selection of reliable and cost-effective safety-critical system architectures," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 10, pp. 2109–2123, 2020.

[12] D. Kirov, R. Passerone, and M. Donelli, "Statistical characterization of the 2.4 GHz radio channel for WSN in indoor office environments," in *Proceedings of the 21st International Conference on Emerging Technologies and Factory Automation*, Berlin, Germany, Sept. 6-9, 2016.