Meeting Stringent QoS Requirements in IIoT-based Scenarios

Published Version:
Herrera, J.L., Bellavista, P., Foschini, L., Galan-Jimenez, J., Murillo, J.M., Berrocal, J. (2020). Meeting Stringent QoS Requirements in IIoT-based Scenarios [10.1109/GLOBECOM42002.2020.9322328].

Availability:
This version is available at: https://hdl.handle.net/11585/797163 since: 2021-03-01

Published:

DOI: http://doi.org/10.1109/GLOBECOM42002.2020.9322328

J. L. Herrera, P. Bellavista, L. Foschini, J. Galán-Jiménez, J. M. Murillo and J. Berrocal, "Meeting Stringent QoS Requirements in IIoT-based Scenarios," GLOBECOM 2020 - 2020 IEEE Global Communications Conference, Taipei, Taiwan, 2020, pp. 1-6

The final published version is available online at ***https://dx.doi.org/10.1109/GLOBECOM42002.2020.9322328***

Meeting Stringent QoS Requirements in IIoT-based Scenarios

Juan Luis Herrera*, Paolo Bellavista†, Luca Foschini†, Jaime Galán-Jiménez* and Javier Berrocal*
*Department of Computer Systems and Telematics Engineering, University of Extremadura, Spain.
†Dipartamento di Informatica-Scienza e Ingegneria, University of Bologna, Italy.
[jlherrerag, jaime, jberolm]@unex.es, [paolo.bellavista, luca.foschini]@unibo.it

***Abstract*—The Industrial Internet of Things (IIoT) provides automation solutions for industrial processes through the interconnection of different sensors, actuators and robotic devices to the Internet, enabling for the automation of manufacturing processes through Factory Automation. However, IIoT processes are often critical, and require very high Quality of Service (QoS) to work properly; as well as network scalability and flexibility. Fog computing, a paradigm that brings computation and storage devices closer to the edge of the network to enhance QoS; as well as Software-Defined Networking (SDN), which enables for network scalability and flexibility through the decoupling of the data and control planes, can be integrated into IIoT architectures in the form of fog nodes that integrate both, computation resources and SDN capabilities, to meet these needs. However, the QoS of the IIoT system depends on the placement of these fog nodes, creating a need to obtain placements that optimize QoS in order to meet the requirements. In this paper, this fog node placement problem is formalized and solved by means of Mixed Integer Programming. We also show relevant experimental results of our formulation and analyze its performance.**

I. Introduction

With the rise of the Internet of Things (IoT) in recent years, the interest on integrating Internet-connected devices into industry to simplify and automate industrial processes has also been increasing, leading to the creation of the Industrial Internet of Things (IIoT) [1]. Some of these IIoT applications, such as Factory Automation (FA), aim at fully automating manufacturing processes. However, IIoT applications such as FA are characterized by having very strict requirements for their Quality of Service (QoS), such as cycle times of 1 millisecond [1].

These QoS requirements call for architectures that are specifically built to support them. Currently, the cloud computing paradigm is extensively used in other IoT fields. However, meeting the strict QoS of critical IIoT applications such as FA can be complicated due to the physical distance between cloud servers and IIoT devices. This has brought paradigms such as fog computing, which brings computation and storage resources closer to IIoT devices, to the scene. Fog computing can be an important enabler for critical IIoT applications [2], mainly to meet response time requirements. Since these fog nodes are closer to end users, data has to go through shorter paths, and thus, it takes less time to transmit.

Another key aspect on IIoT is scalability. In order to support FA, the network needs to account for scalability and flexibility, while still delivering a high enough QoS to meet its requirements [1]. In this respect, the Software-Defined Networking (SDN) paradigm provides a decoupling between the data and the control planes, which enables high flexibility and scalability in these networks and can play a critical role in IIoT [1], [3].

It is because of this that architectures that combine both fog computing and SDN are a key enabler for IIoT, allowing to meet the stringent QoS requirements of FA while delivering flexibility and scalability. In [4], some of the authors of this work presented a proposal for a self-adaptive framework that combines both fog and SDN aimed at IIoT, as well as for a specific Fog Node (FN) for the architecture. The FN includes an SDN switch and a virtualization platform to provide IIoT services. The FN is designed to host IIoT services for applications such as FA to meet its stringent QoS requirements, and thus allows for an integration of fog computing and SDN aimed at IIoT QoS-strict applications.

However, as also tested in [4], the placement of this FN impacts on the QoS of the architecture. While these tests are limited to two scenarios, i.e. deploying the FN in the local network and deploying it in a remote network, it sets the idea that the placement of the FN can affect the QoS of IIoT applications. Therefore, if these IIoT applications have strict QoS requirements, such as FA, it is key to place the FN in such a way that QoS is optimized. Otherwise, the possibility of using QoS-strict IIoT applications could be at risk.

While optimizing the placement is a way to optimize QoS, it is not always possible to make the distance between a single FN and IIoT devices short enough to meet the QoS needs of FA. In these cases, the placement of more than one FN has to be optimized. Moreover, there is also a need to optimize the routes between FNs and IIoT devices, so that each IIoT device consumes the services of the FN that provides the best QoS to it. These routes can then be implemented in the network using SDN, allowing for a transparent solution for FA and other IIoT applications [3]. The problem of placing a certain number of FNs optimally in the network, finding which FNs provide the best QoS to which IIoT devices and finding the optimal routes between those, so that QoS is maximized, is what we label the Fog Node Placement Problem (FNPP). In this work, we provide a solution for the FNPP that, when applied on a certain architecture, is able to find all these results.

The main contributions of this paper are:

- The formalization of the FNPP.

- The formulation of the solution to the FNPP using Mixed Integer Programming (MIP).
- An analysis of the results of applying the solution to architectures under different circumstances.

The remainder of this paper is structured as follows. Section II motivates the FNPP. Section III explains the system model we will use in the FNPP. Section IV includes the problem formulation as a MIP optimization problem in order to find its solution. Section V presents details of our test environment and experimental results. Finally, section VI concludes the paper.

II. Motivation

The FNPP is mainly motivated by IIoT applications that have very stringent QoS requirements, such as FA. In these applications, functionality is inherently related to QoS, and the QoS to meet is very strict, e.g. 1 ms response time [1]. To this respect, it is crucial to minimize response time in order to meet this requirement.

There are two methods to improve response time of a system, either the computing QoS (i.e. execution time) or the networking QoS (i.e. latency) have to be improved [5]. Fog computing provides a solution that enables for both, bringing computation resources closer to IIoT devices to provide them with services that require very low latency [2]. Moreover, the SDN paradigm provides scalability and flexibility to the network, two key requirements of FA [1]. Therefore, SDN and fog computing should be integrated in IIoT scenarios to meet the needs of FA.

The proposal in [4], namely the FN, enables for the integration of fog computing and SDN, while being designed specifically for IIoT. FNs can therefore meet both necessities of FA, by providing IIoT services close to the IIoT devices that consume them as well as enabling for a flexible and scalable network by including an SDN switch. However, the placement of FNs is key for the QoS of the IIoT services they host, with a sub-optimal placement being able to put the feasibility of a FA system at risk, i.e. the solution to the FNPP is key to meet the QoS needs of FA.

Although the need for QoS, flexibility and scalability of FA systems and similar IIoT applications, manifested in the form of a need for fog computing and SDN, can be solved by using FNs; this creates a need for optimizing their placement. This is the need that mainly motivates this paper. Therefore, the main goal of this paper is to provide a solution to the FNPP in form of a MIP formulation to optimally assess the placement of FNs in a specific IIoT scenario.

To the best of our knowledge, no prior works tackle the problem of placing a set of fog nodes in such a way so that QoS is optimized. However, both the optimization of QoS through the placement of equipment and the optimization of fog infrastructures in different ways are currently active research topics [6]–[8].

On the one hand, different works have conducted research on optimizing the placement of a SDN controller to optimize the QoS of a network [6]. On the other hand, optimization of QoS in fog infrastructures is an open topic in research. [7] proposes an optimization based on a placement of services in fog infrastructures that violates QoS requirements as least as possible. [8] optimizes QoS by finding the optimal node in a fog infrastructure to execute a certain service dynamically and to integrate these optimal solutions with a SDN controller so they are applied transparently.

The main differences between the works presented and this work are mainly related to their focus. The works related to SDN controllers aim at optimizing the QoS of the network, as opposed to optimizing the QoS of IIoT applications instead. On the other hand, related works on the optimization of fog infrastructures are focused on placing services in an infrastructure of already placed FNs, not at placing the FNs themselves.

III. System model

In this section, the FNPP is explained in detail with an example model. For this model, a FA IIoT application for automated welding will be considered. In this application, each smart welder constantly senses the item in front of it. If it is a piece of metal, it notifies an smart welding service to retrieve a command on how to weld it. In this particular example, the system is comprised of four smart welders and four SDN switches, connected in a topology as shown in Figure 1.

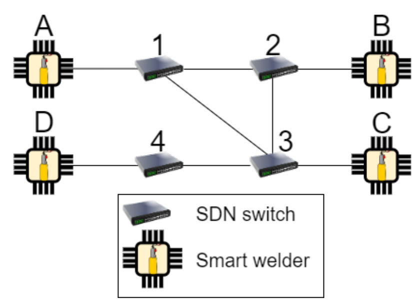

Figure 1. Example system model.

Since the smart welders have to quickly perform cycles in which they sense metal, send the information to the service and retrieve commands, the smart welding service runs on a FN to meet its strict latency requirements. This FN contains a SDN switch while also hosting IIoT services, in this case, the smart welding service. Therefore, one of the SDN switches shown in Figure 1 should be replaced by a FN. However, any of the shown switches could potentially be replaced by a FN. To meet the strict latency requirement, the FN cannot be more than one hop away from a smart welder. Therefore, there is a need to solve the FNPP in order to meet the requirement.

If switch 1 were to be replaced by a FN, A, B and C would meet the requirement. D, however, would be two hops away, and thus, it would not meet the requirement. Similar results are obtained if the FN replaces switches 2 or 4. If it were to replace switch 3, however, it would be possible to meet the latency constraint, given that A would not route through switch 2. Therefore, it is needed to obtain the optimal placement for the FN and the optimal routes to be followed by the traffic of every smart welder in order to meet the stringent QoS requirement that has been set.

Another approach could be replacing two switches by FNs instead. In that case, some smart welders would consume the service from one FN, while the rest of smart welders would consume it from the other one. In that case, if one of the FNs is placed on switch 3 and another one is placed on switch 1, D should consume the service from the FN on switch 3. If it were to consume it from switch 1, even with an optimal route, it would be two hops away. Therefore, when multiple FNs are considered, not only the placement of the FNs and the routes followed by the traffic are important, the mapping between IIoT devices and FNs is too.

While the FNPP can be solved manually in small cases like these, scalability is crucial in IIoT [1]. When these architectures grow, testing every single placement, route and mapping manually can become unbearable in terms of time.

IV. Problem formulation

In order to solve the FNPP in larger architectures, we present a formulation using MIP, that can be used with automatic MIP solvers to solve the FNPP.

To define the FNPP, the network topology is represented as a directed graph $G = \{V, L\}$, where V are the vertices and L are the links. A link $l_{ij} \in L; i \neq j; i, j \in V$ links together vertices i and j. Let C_{ij} be the capacity of the link l_{ij}, so that $C_{ij} = 0 \forall l_{ij} \notin L$.

We also make certain assumptions on the network topology. Namely, we assume that:

- The vertices are either SDN switches or IIoT devices.
- The amount of SDN switches that should be replaced by FNs is given as input, and should be 1 or greater.
- All FNs have the same capacity.
- The combined capacity of all FNs is enough to deal at least with all the traffic in the network, and each FN has enough capacity to deal at least with the traffic from a single host.

With these assumptions in mind, let $H \in V$ be the set of vertices that are hosts and $S \in N$ be the set of vertices that are switches. Thus, $V = H \cup S$ and $H \cap S = \emptyset$. S will therefore be the set of possible placements for FNs. Similarly, every host $h \in H$ generates an amount of traffic $\phi_h \geq 0$ that has to be processed by its mapped FN. A FN has the capacity to process up to α traffic per unit of time.

Let $P(i, j) = \{L', V'\}$ be the shortest path from vertex i to vertex j, which traverses vertices V' and links L', so that $L' \subseteq L; V' \subseteq V$. Let $D(P(i, j)) = |V'| - 1$ be the amount of vertices traversed in path $P(i, j)$ (e.g. a path that directly connects i with j would have $D(P(i, j)) = 1$). Let each link have a propagation latency of $\beta_{l_{ij}} \geq 0$ and every switch have a processing latency of $\beta_S \geq 0$. Let $L(i, j) = \sum_{l_{ab} \in L'} \beta_{l_{ab}} + (D(P(i, j)) - 1)\beta_S$ be the latency of sending traffic from vertex i to vertex j.

Let θ be the upper limit for the number of FNs to be placed. Concerning the set of variables, let $X_i, i \in V$ be a binary variable that will be 1 if a FN is placed on vertex i and let $Y_{ij}, i, j \in V$ be a binary variable that will be 1 if vertex i is *mapped* to the FN placed in j. Finally, let f^h_{ij} be a binary variable that will be 1 if the traffic generated by h is routed through the link l_{ij}.

The set of notations is summarized in Table I.

Parameter	Meaning
G	Graph that represents the network
L	Set of links of the network
V	Set of vertices of the network
H	Set of hosts (i.e. IIoT devices) of the network
S	Set of SDN switches of the network
C_{ij}	Capacity of link l_{ij}
ϕ_h	Traffic generated by host h
α	Maximum traffic that can be processed by a FN per unit of time
$P(i,j)$	Shortest path from vertex i to vertex j
$\beta_{l_{ij}}$	Propagation latency of link l_{ij}
β_S	Processing latency of a SDN switch
$L(i,j)$	Latency of traffic sent from vertex i to vertex j
θ	Maximum number of FNs to be placed
Variable	**Meaning**
X_i	Boolean to determine if a FN is placed in vertex i
Y_{ij}	Boolean to determine if vertex i is mapped to the FN located in vertex j
f^h_{ij}	Boolean to determine if traffic generated by host h is routed through link l_{ij}

Table I
List of notations

Then, the FNPP solution can be formulated as follows:

$$\min \sum_{i \in V} \sum_{j \in V} L(i,j) Y_{ij} \quad (1)$$

subject to:

$$i \in V, h \in H : \sum_{j \in V} f^h_{ij} - f^h_{ji} = \begin{cases} 1 & \text{if } i = h \\ -Y_{hi} & \text{otherwise.} \end{cases} \quad (2)$$

$$\forall l_{ij} \in L : \sum_{h \in H} f^h_{ij} \phi_h \leq C_{ij} \quad (3)$$

$$\sum_{i \in V} X_i \leq \theta \quad (4)$$

$$\forall i \in V : \sum_{h \in H} \phi_h Y_{hi} \leq \alpha X_i \quad (5)$$

$$\forall h \in H : \sum_{i \in V} Y_{hi} = 1 \quad (6)$$

$$\forall s \in S : \sum_{i \in V} Y_{si} = 0 \quad (7)$$

$$\forall h \in H : X_h = 0 \quad (8)$$

$$\forall i, j \in V, h \in H, l_{i'j'} \in L : X_i, Y_{ij}, f^h_{i'j'} \in \{0, 1\} \quad (9)$$

Equation 1 expresses the optimization objective, i.e. to minimize the sum of the latencies from each host to its mapped FN. Equation 2 represents an adaptation of the classical flow conservation constraint to this scenario, while Equation 3 constrains the total traffic routed by a link to be less than its capacity. Equation 4 constrains the number of FNs: there must not be more than θ. Equation 5 controls the capacity of a FN, so the amount of traffic directed to that FN is never more that it can handle. Equations 6 and 7 will make sure that a host is mapped to exactly one FN and that switches are mapped to no FNs. Finally, Equation 8 assures no FNs will be set up on host vertices. Equation 9 simply makes these variables binary.

This mathematical formulation represents the FNPP, allowing it to be sovled in different network topologies with different characteristics for the FN or the IIoT devices, providing the optimal placement, routes and mapping to meet the strict latency requirements of IIoT applications such as FA.

V. Performance evaluation

In this section, a performance evaluation of our solution to the FNPP to analyze its performance under different circumstances is presented. Analyses of latency varying the number of FNs placed, traffic, number of hosts in the network and the criteria used to place FNs have been performed to test the latency reduction of our formulation. Analyses of link load varying the amount of hosts in the network have also been performed to test its impact.

A. Evaluation environment

The formulation has been evaluated in four different topologies to analyze scalability and execution time. Namely, we have used small (12 switches, 15 links), medium (17 switches, 26 links), large (22 switches, 36 links), and extra-large (50 switches, 88 links) SDN topologies, generated synthetically.

We have considered different parameters for these simulations. Parameter α is the capacity of the FN in Mbps. Since different topologies have different amounts of traffic, and thus require different values for α, the comparisons for multiple topologies have considered α expressed as a percentage of the total traffic in the network instead of in Mbps. It is important to keep in mind that not every value for α is valid, since there must be enough overall capacity to process all the traffic on the network, as well as to have enough capacity on a FN to, at least, process all the traffic coming from a single node, as Equation 10 shows.

$$\alpha \geq max(max(\phi_h \forall h \in H), \frac{sum_{h \in H} \phi_h}{\theta}) \tag{10}$$

Parameter β_S is the processing latency of a SDN switch in milliseconds. We have considered three values for it, concretely $\beta_S \in \{0.15, 0.76, 2.21\}$, retrieved from [9].

In order to evaluate the impact of the number of FNs to be placed in the network and the amount of hosts, the θ parameter as described in Section IV and a γ parameter that sets the amount of hosts per switch have been considered. The traffic on the network does not change regardless of γ, and is thus divided equally between the hosts of each switch. Traffic analyses have been performed by multiplying this traffic by factors between 0.5 and 1.0.

B. Performance analysis

The first analysis that is proposed aims at evaluating the average latency on the medium topology varying θ and γ, while each fog node can process all the traffic in the network, as Figure 2 shows. When θ increases, latency lowers in a negative log likelihood. This is because, when more FNs are placed, they can generally be closer to their mapped hosts, thus reducing latency, until they reach the switch they use to access the network. Once there, it is not possible to place FNs any closer. Therefore, the latency decrease is very significant when θ is low and FNs are in intermediate positions. However, it gets less steep as θ increases and FNs are closer to hosts, thus the negative log likelihood. As for the number of hosts per switch, it slightly influences latency, and its influence is not greatly affected by θ.

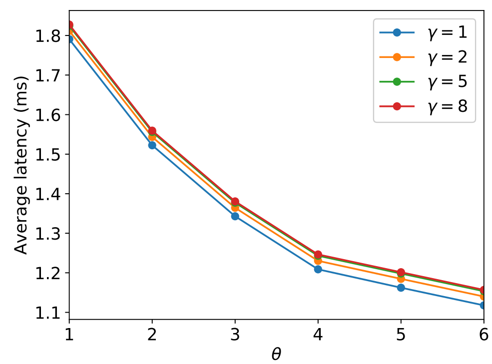

Figure 2. Average latency vs θ in medium topology.

The second analysis aims at evaluating the average latency on the medium topology varying the traffic load, as Figure 3 shows. In this analysis, the values for FN capacity and θ have been experimentally established in such a way that FNs are stressed when traffic rises. As we can see, when the network is not heavily loaded (i.e. traffic scaling between 0.5 and 0.9), latency is stable. However, in stress situations (i.e. traffic scaling of 1,0), there is a latency spike. This is because some areas produce more traffic than others, and thus, once the closest FN to the area is at its full capacity, the remaining hosts in that area have to be mapped to a FN in another area, and thus, further away, increasing the latency for these hosts. The main difference γ makes is about mapping few IIoT devices with high resource requirements or many IIoT devices with low resource requirements, which influences how many hosts have to be mapped to other areas and how they are mapped.

Figure 4 the average latency on the medium topology as a function of α. The objective of this analysis is to evaluate the average latency varying the FN capacity. It is clear that the number of hosts per switch affects the effects of α: the higher γ is, as long as it is larger than 1, the less steep the latency reduction is. On the other hand, it is also shown that a higher α generally improves latency. This is because of the effect commented earlier: a higher α makes each area be less

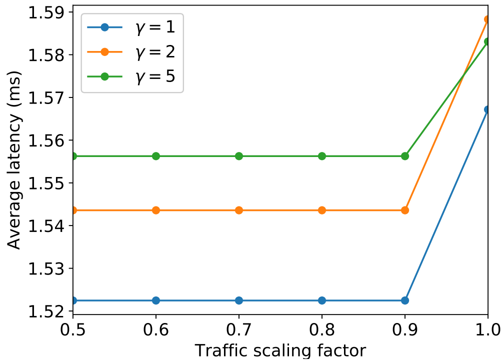

Figure 3. Average latency vs traffic in medium topology.

overloaded with traffic, and thus, it makes the hosts in said area able to direct all their traffic to the FN in their area. The higher α is, the less traffic has to be directed to FNs in areas further away. However, each host can only direct its traffic to a single FN, hence, $\gamma > 1$ improves this effect, as different hosts can be mapped to different FNs. It can also be seen that lower γ values have a lower latency with low values of α but a higher latency with high values of α, compared to other values of γ.

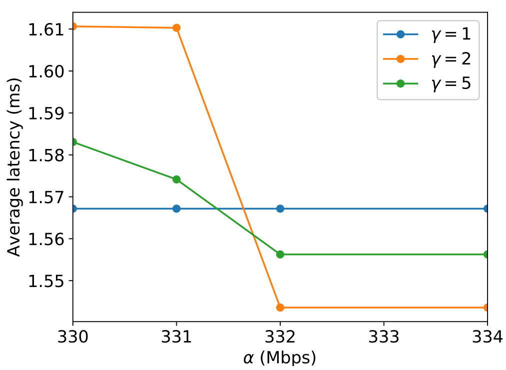

Figure 4. Average latency vs α in medium.

In order to evaluate the scalability of our proposal, the next analysis evaluates the link load in different topologies. Figures 5, 6 and 7 show the empirical CDF of the link load in different topologies. The size of the topology and the number of hosts per switch are both key to this respect. In general, it can be seen that the links of larger topologies are less loaded than those of smaller topologies. These patterns are also replicated when considering γ: a higher γ makes the CDF steeper. γ also has a major role on the number of unused links: a lower γ leaves more links unused. One of the main reasons for this is that a higher γ adds an additional link for each host, that will always be used. Thus, these additional links make the proportion of overall unused links lower.

Figure 8 shows a comparison between choosing the placement of the FNs by using different graph theory-based techniques and choosing the placement by using the formulation proposed in Section IV, labeled as *Optimal*. In all of the cases, the host-FN mapping and the traffic routing were performed by means of the formulation in order to minimize latency. As Figure 8 shows, the minimum latency achieved by each technique is quite different when $\theta = 1$. However, the performance gap shrinks when θ is larger. In any case, placing the FNs in the nodes with the highest betweenness centrality gives the best results out of the three techniques. Nonetheless, there is a large performance gap between using any of these techniques and using the proposed formulation, although it shrinks when θ is large enough. This is because, when θ is larger, more FNs can be placed, and thus, it is more likely to place FNs at their optimal placements.

Finally, Tables II and III show the results of an analysis to evaluate the scalability of our solution on large networks based on the mean execution time of the simulations that considered our formulation. The results for the medium topology can be seen in Table II. In the medium topology, the MIP solver usually takes more time to solve the problem when $\theta = 2$, while it usually manages to keep similar times in the rest of cases, taking into account that γ is the most important parameter that influences execution time, since more hosts per switch means an overall larger topology, and, as it can be seen in Table III, the size of the topology is a key parameter on execution time. As for the times themselves, the worst time is for solving the FNPP in the medium topology with $\gamma = 8, \theta = 5$ takes roughly over 3 minutes. On the other hand, in smaller topologies, such as the small or medium topology with $\gamma = 1$, the FNPP can be solved in about one second. In the middle ground, it takes roughly 25 seconds to solve the FNPP in the extra-large topology.

Table II
AVERAGE EXECUTION TIMES OF THE SIMULATIONS IN MEDIUM TOPOLOGY

γ	θ	Time (s)	γ	θ	Time (s)
1	1	0.708	5	1	36.069
	2	1.680		2	35.239
	3	0.878		3	30.325
	4	0.830		4	30.320
	5	1.139		5	32.957
	6	0.953		6	31.205
2	1	3.490	8	1	136.489
	2	7.582		2	143.020
	3	3.017		3	138.938
	4	2.996		4	128.652
	5	3.584		5	189.306
	6	2.906		6	135.589

Table III
AVERAGE EXECUTION TIMES OF THE SIMULATIONS IN DIFFERENT TOPOLOGIES

Topology	$\lvert V \rvert$	$\lvert L \rvert$	Time (s)
Small	24	27	0.810
Medium	34	43	1.912
Large	44	58	1.902
Extra-large	100	138	25.100

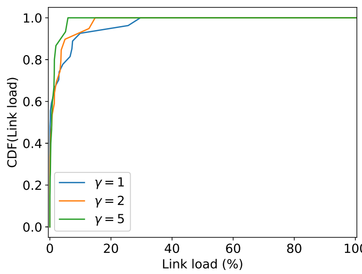

Figure 5. Empirical CDF of link load in small topology.

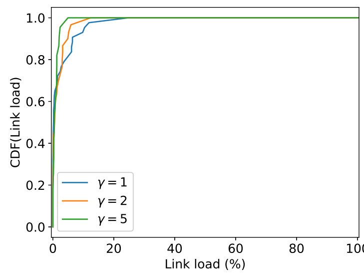

Figure 6. Empirical CDF of link load in medium topology.

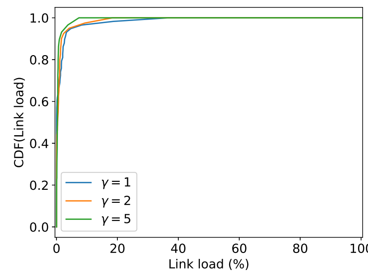

Figure 7. Empirical CDF of link load in large topology.

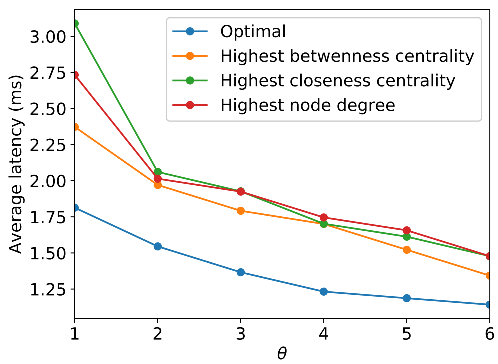

Figure 8. Average latency using different solutions in medium topology.

In the IIoT domain, these results show that a high amount of FNs is crucial to minimize latency, and that assessing their placement, mapping and routing by using our proposal can further minimize this latency. By repeatedly solving the FNPP and installing its routes and mappings on the SDN controller, it is possible to adapt the placement over time to minimize latency, all while being transparent to the IIoT devices.

VI. Conclusions

In this paper, we have introduced, formulated and proposed a solution for the problem of placing one or more fog nodes to optimize the latency of IIoT applications, i.e. a solution for the FNPP. Furthermore, we have tested the solution formulation on different network topologies with different parameters and we have analyzed the effects of these parameters in network latency. We have also tested the scalability of our solution in different circumstances.

Our formulation is able to optimize the placement and mapping of FNs, as well as to optimize the routing of the traffic that each node offloads to its fog node in tractable time. Therefore, it is possible to use the proposed solution over time, thus adapting FN placement, mapping and routing to fit dynamicity. The speed of these adaptations is dominated by execution time, and thus, it is not possible to adapt the placement, mapping and routing in larger networks in less than a few minutes.

In the future, we expect to analyze the impact of adding FNs in terms of cost, as well as to develop heuristics to improve scalability and execution time, so this problem can be solved in a shorter time in large networks, and thus its adaptation time can be shortened. We also expect to evaluate the performance of our formulation and these heuristics in real or emulated network test-beds.

VII. Acknowledgements

This work has been partially funded by the project RTI2018-094591-B-I00 (MCI/AEI/FEDER,UE), the 4IE+ Project (0499-4IE-PLUS-4-E) funded by the Interreg V-A España-Portugal (POCTEP) 2014-2020 program, by the Department of Economy and Infrastructure of the Government of Extremadura (GR18112, IB18030), and by the European Regional Development Fund.

References

[1] H. Xu, W. Yu, D. Griffith, and N. Golmie, "A Survey on Industrial Internet of Things: A Cyber-Physical Systems Perspective," *IEEE Access*, vol. 6, pp. 78 238–78 259, 2018.

[2] M. Aazam, S. Zeadally, and K. A. Harras, "Deploying fog computing in industrial internet of things and industry 4.0," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4674–4682, 2018.

[3] J. Wan, S. Tang, Z. Shu, D. Li, S. Wang, M. Imran, and A. V. Vasilakos, "Software-defined industrial internet of things in the context of industry 4.0," *IEEE Sensors Journal*, vol. 16, no. 20, pp. 7373–7380, 2016.

[4] I. Bedhief, L. Foschini, P. Bellavista, M. Kassar, and T. Aguili, "Toward self-adaptive software defined fog networking architecture for IIoT and industry 4.0," in *Proceedings of IEEE CAMAD 2019*, 2019.

[5] ITU-T, *Definitions of terms related to quality of service*, International Telecommunication Union Telecommunication Standarization Sector Std. E.800, Rev. 09/2008.

[6] T. Das, V. Sridharan, and M. Gurusamy, "A Survey on Controller Placement in SDN," *IEEE Communications Surveys & Tutorials*, pp. 472–503, 2019.

[7] O. Skarlat, M. Nardelli, S. Schulte, and S. Dustdar, "Towards QoS-Aware Fog Service Placement," in *Proceedings of IEEE ICFEC 2017*, aug 2017, pp. 89–96.

[8] R. P. Singh, J. Grover, and G. R. Murthy, "Self organizing software defined edge controller in IoT infrastructure," in *ACM International Conference Proceeding Series*, 2017, pp. 1–7.

[9] P. Emmerich, D. Raumer, S. Gallenmüller, F. Wohlfart, and G. Carle, "Throughput and Latency of Virtual Switching with Open vSwitch: A Quantitative Analysis," *Journal of Network and Systems Management*, vol. 26, no. 2, pp. 314–338, 2018.