MULTI-FACETED MODELING IN THE ANALYSIS AND OPTIMIZATION OF IOT COMPLEX SYSTEMS

Román Cárdenas
Patricia Arroba
José M. Moya

Laboratorio de Sistemas Integrados (LSI)
CCS-Center for Computational Simulation
Universidad Politécnica de Madrid
ETSI Telecomunicación, Avenida Complutense 30
Madrid 28040, Spain
{r.cardenas,p.arroba,jm.moya}@upm.es

José L. Risco-Martín

Dpt. of Computer Architecture and Automation
CCS-Center for Computational Simulation
Universidad Complutense de Madrid
C/ Prof. José García Santesmases, 9
Madrid 28040, Spain
jlrisco@ucm.es

ABSTRACT

Modeling, Simulation, and Optimization tools are used in the development process of complex systems for improving their overall performance. However, enforcing full coverage in the validation process of these models may add unbearable simulation time overheads. In this research, we propose the implementation of multi-faceted sets of models with different degrees of complexity for optimizing IoT complex systems. With this approach, elements of the system are optimized using simpler and faster models that reduce simulation times, enabling the exploration of more scenarios in less time. Optimized components can be then validated extensively using more complex models, ensuring full coverage, and reducing potential failures.

Keywords: Systems of Systems, Multi-Resolution Modeling, Optimization, Performance.

1 INTRODUCTION AND RELATED WORK

Developing complex systems often requires the definition of multi-domain solutions. For example, an Internet of Things (IoT) application may require telecommunication infrastructures, the development of firmware for embedded systems, security mechanisms, the hosting of computing resources in the Cloud, and the integration of all these elements (Karbalaei et al. 2018). In this multidisciplinary scenario, communication between the different teams working on the system's development arises as one of the principal sources of failures during the system conception process. For overcoming this vulnerability, Model-Based Systems Engineering (MBSE) methodologies establish modeling as the central entity for exchanging information (Russell 2012). MBSE enables the use of Modeling and Simulation (M&S) tools to explore the complex system under study and validate the proposed solution before its implementation, assessing technical risks and functionality of the solution while reducing both operational and capital expenses (Mittal and Tolk 2020).

In recent years, the system development process has integrated M&S for comparing different design alternatives and selecting the one that presents better results (de Sousa Junior et al. 2019). Modeling, Simulation, and Optimization (M&S&O) tools enable the optimization of the system under study, making extensive usage of M&S. In this context, we developed Mercury (Cárdenas et al. 2020), an M&S&O framework for

Fog Computing infrastructures (Yuan et al. 2018). Mercury defines a versatile, highly detailed model of the principal entities present on Fog Computing scenarios, enabling the exploration of different communication protocols, network topologies, routing algorithms, and computing resource management algorithms.

However, complex systems often imply complex models for validating the proposed solution. In these cases, using a single, monolithic M&S&O tool for optimizing any part of the system may add a high simulation time overhead, being unbearable to make an in-depth exploration of all the possible scenarios. In case of Mercury, each simulation required for obtaining the results in Cárdenas et al. 2020 took more than 10 hours.

From the use case perspective, a common approach for reducing simulation overheads is the development of models with simpler behavior that remove those parts of the complex system that are not directly related to the elements of the system under study. For instance, in state-of-the-art Fog Computing simulators, physical means used by nodes to communicate with each other, network access control policies, resource discovery protocols, and dynamic network routing algorithms are usually not considered (Gupta et al. 2017, Sonmez et al. 2018). On the other hand, from the M&S perspective, several research work is focused on reducing the complexity of models while keeping the behavior of its elements intact. For example, when dealing with models that make use of stacked protocols for communication, it is a common practice to implement shortcuts that skip the lowest layers of the stack (Dalle, Zeigler, and Wainer 2008). While these approaches reduce simulation times, they also provide less accurate results, and do not ensure full coverage of the solution, as they remove part of the relationships between the elements that conform the complex system.

In this research, we propose a multi-faceted M&S&O workflow for Mercury. This approach defines a set of models with variable degree of detail. Thus, when optimizing a feature of the system under study, the optimization is performed against a model with fine-grained level of detail in the parts of the system that affect the most this feature but a high-level model of the rest of the system. Once the feature is optimized, the decision is extensively validated on the detailed model of the whole system with less effort, since the design space has been already explored and exploited. Doing so, each part of the system is tested and optimized ensuring full coverage and reducing system development time.

The paper is organized as follows. In Section 2, we present Mercury, and provide an overview of its original implementation. Section 3 presents a set of simplified versions of Mercury with the aim of optimizing computation offloading, without taking into consideration other features such as inter-node communication nor network access control issues. Section 4 compares the output provided by these simpler models with the original detailed model, and estimates the complexity reduction of each alternative solution. Conclusions and future work are outlined in Section 5.

2 BACKGROUND: MERCURY

Mercury is an M&S&O framework for Fog Computing infrastructures. The conception of Mercury is to provide a fine-grained level of detail of each element that conforms the Fog Computing scenario with the aim of assisting on the decision-making process of the tasks of designing and operating Fog Computing infrastructures. Figure 1 shows an schematic of the components of Mercury. The structure and behavior of the model is briefly explained below. For an in-depth description, refer to our previous research (Cárdenas et al. 2020).

The **IoT devices layer** contains all the end users (i.e., User Equipment(UE)). UE devices are embedded systems with limited processing and energy resources (e.g., smartphones). These embedded systems use computation offloading to delegate data processing that requires complex, energy-demanding operations (e.g., training a Deep Neural Network (DNN) in incremental learning applications (Gepperth and Hammer 2016)) to a more powerful computing node. The **edge federation layer** contains the Edge Data Centers (EDCs) that comprise the federated pool of resources used by UE for computation offloading. When re-

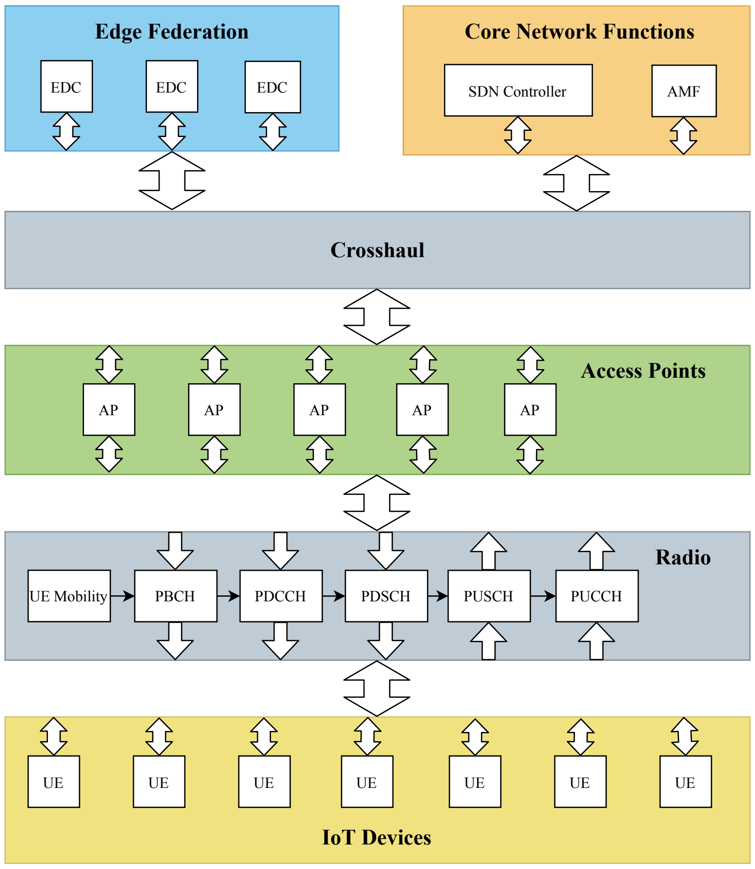


Figure 1: Original Mercury architecture.

quested by UE, the EDC that receives the computation offloading request starts a session. A session consists of reserving computation resources for processing incoming data from UE, thus granting computation offloading availability until the end of the session. When a session is closed, resources are again available to eventual new computation offloading requests. UE that wants to use computation offloading must be connected to the Radio Access Network (RAN). The **Access Points (APs) layer** models all the radio nodes installed by the Internet Service Provider (ISP) for providing access to the RAN. The **Core Network Functions (CNFs) layer** models all the infrastructure management functions performed by the ISP that owns the network. In particular, two of these functions are modeled: the Access and Mobility Management Function (AMF), which processes new connection requests from UE to ensure that they have permission to connect and keeps track of which UE device is connected to each AP, and the Software-Defined Network Controller (SDNC), which is in charge of routing incoming computation offloading requests to the most suitable EDC of the federation. The **crosshaul and radio layers** model communication networks used by the rest of the elements to communicate with each other. The crosshaul network is modeled as a fully connected fiber optics mesh network. On the other hand, the radio network is composed of five independent, unidirectional radio channels: the Physical Broadcast Channel (PBCH) is used by APs to broadcast signaling messages to UE; the Physical Downlink Control Channel (PDCCH) and Physical Uplink Control Channel (PUCCH) are used by APs and UE, respectively, for sending connectivity-related messages (e.g., access, handover and disconnections from the RAN); the Physical Downlink Shared Channel (PDSCH) and Physical Uplink Shared Channel (PUSCH) are used for sending messages related to computation offloading processes. Additionally, the radio layer manages UE mobility.

Crosshaul and radio layers impose propagation and transmission delays to messages going through them:

$$D = D_{trans} + D_{prop} = \frac{s}{C} + \frac{d}{V_{prop}} \tag{1}$$

where s is the size of the message (in bits), C is the link capacity (in bits per second), d is the distance of the link (in meters), and V_{prop} is the propagation speed of the link (in meters per second). Each channel is modeled as a set of point-to-point links between transmitter nodes and receiver nodes. Mercury computes the capacity of a link as the maximum theoretical link capacity according to the Shannon-Hartley theorem (Shannon 1948):

$$C = B \cdot \log_2 (1 + SNR) \tag{2}$$

where SNR is the Signal-to-Noise Ratio (SNR) of the link (i.e., link budget). Figure 2 shows an schematic of the link budget model implemented in Mercury.

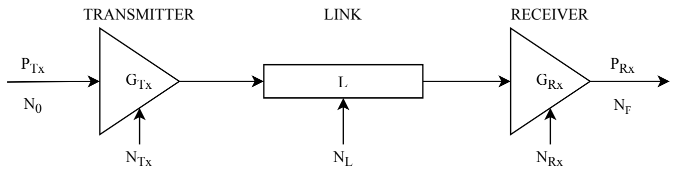


Figure 2: Mercury link budget model.

The link budget is computed according to the Friis transmission equation (Shaw 2013):

$$SNR = \frac{P_{Rx}}{N_F} = \frac{P_{Tx} \cdot G_{Tx} \cdot L \cdot G_{Rx}}{((N_0 \cdot G_{Tx} + N_{Tx}) \cdot L + N_L) \cdot G_{Rx} + N_{Rx}} \tag{3}$$

Every message from node A to node B contains an initial power P_{Tx} and noise power N_0. The transmitter of node A amplifies both with a gain G_{Tx} and introduces additional noise N_{Tx}. Depending on the distance between the nodes, both power and noise are attenuated according to a path loss function. Links may introduce additional noise N_L due to interference or other physical phenomena. The receiver node B amplifies incoming messages with a gain G_{Rx} and introduces additional noise N_{Rx}.

Mercury is a fine-grained M&S&O tool that allows to explore multiple features of the scenario under study. It enables the user to compare different computation resources allocation strategies of the edge federation to study their impact on the power consumption of the infrastructure and the delay perceived by end users. For connectivity, it models a detailed 5G RAN, with AMF for access control and SDNC for network routing. Communication protocols are based on the Release 15 of the 3rd Generation Partnership Project (3GPP 2019). Furthermore, it is possible to explore different physical communication interfaces (e.g., 5G New Radio physical protocol) to see their impact on the quality of service.

3 MULTI-FACETED COMPUTATION OFFLOADING MODELS IN MERCURY

Even though Mercury provides a very versatile tool set for exploring cutting-edge technologies, our research line is mainly focused on workload allocation and energy optimization of the edge federation, and other aspects of this complex system were less important for us. However, when exploring different scenarios, we experienced very long simulation times. Therefore, towards exploring more advanced resource management configurations, we faced the need of implementing alternative, simpler models for speeding up simulation results. This section presents a set of multi-faceted models for Mercury that, while keeping a fine-grained level of detail regarding computation offloading processes, introduce lighter models of the rest of entities of the Fog Computing complex systems. Thus, simulation time is reduced when optimizing the elements involved in the computation offloading. The original, highly detailed model is used then to validate optimized outcomes and analyze the effect of other elements that compose the complex system under study, ensuring full coverage of the solution.

3.1 Model with Bypassed Physical Layer

The main components of Mercury (i.e., UE, APs, EDCs, and CNFs) communicate with each other using a layered protocol stack. Mercury defines three different layers: application (e.g., network connectivity or computation offloading), network (end-to-end message passing), and physical (point-to-point communication). Figure 3 shows the communication flow of computation offloading-related application messages.

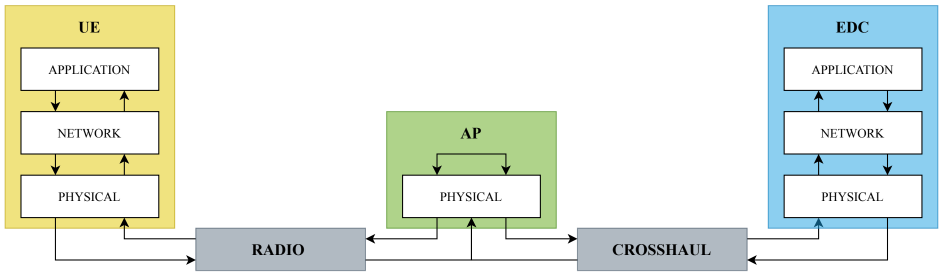


Figure 3: Communication flow of computation offloading-related messages in Mercury.

Application messages from UE are encapsulated into network messages, specifying the EDC in charge of performing the computation offloading as the receiver. This network message is then encapsulated into a physical message and sent to the AP that provides connectivity to the UE device. APs remove the physical layer of incoming messages and re-route the network message to the corresponding EDC after encapsulating it into a new physical message via the crosshaul network.

The model with bypassed physical layer is based on the shortcut modeling pattern for establishing shortcuts between components of a complex model to reduce the overall complexity while keeping the rest of the elements intact (Dalle, Zeigler, and Wainer 2008). Figure 4 shows the topology of this model.

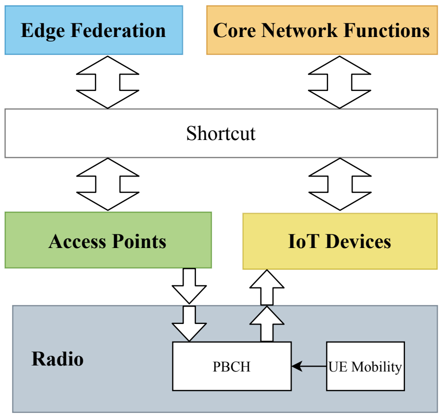


Figure 4: Mercury model with bypassed physical layer.

The behavior of all the principal models remains intact. The main difference with the original model is the lack of a physical layer in the architecture. The crosshaul layer and most of the channels of the radio layer are removed from the architecture. The shortcut module routes network packets from any component to its corresponding receiver. Thus, as shown in Figure 5, in this model the communication flow between computation offloading-related application messages does not go through any AP, as communication is established in an end-to-end manner.

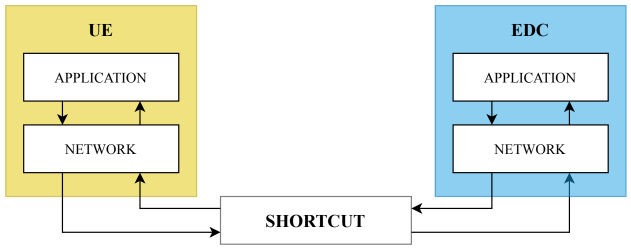


Figure 5: Bypassed Mercury communication layers.

Note that the PBCH channel of the radio layer is still present in the model. This is necessary in order to enable the discovery of the most suitable AP for each UE device: signaling messages of APs go through this physical channel, applying the respective noise and attenuation. When received by UE, these messages provide information about the SNR perceived by the end users, enabling them to select the most suitable AP as gateway.

The simplification of this model remains in the interconnection of the principal components of the scenario. However, these components behave exactly the same as in the original model. As physical networks are removed, no transmission nor propagation delay is applied to messages in the bypassed model. The delay only depends on the edge federation resource management techniques. Therefore, if the user of Mercury is only interested on optimizing EDC resource allocation techniques or Software-Defined Network (SDN) algorithms for setting routing paths between APs and EDCs, this simplified version of Mercury keeps enough level of detail.

On the other hand, as the PDSCH and PUSCH radio channels are removed, it is not possible to analyze the UE bandwidth share: if the user of Mercury wants to detect hot spots in the RAN with the aim of installing the APs in the most suitable locations, this model will not provide enough information. Furthermore, removing the crosshaul network makes impossible to detect any bottleneck in the optical network.

3.2 Simplified Model

In Mercury, there are two main applications from the end user side: computation offloading and network connectivity. In the first application, the main implied components are UE (i.e., computation offloading demand) and EDCs (i.e., computation offloading resources). The SDNCs monitors the status of the edge federation and routes UE requests to the most suitable EDC depending on the location of UE, the availability of resources on the EDC side, or the priority of the requested service, for instance. However, UE must be connected to the RAN to use computation offloading. Therefore, in the connectivity side, the APs serve as gateways for UE, and the AMF is in charge on ensuring that connected UE is allowed to be connected by the ISP. Furthermore, as UE changes their location, they may detect that the AP that provides to them network connectivity is no longer the most suitable one. This may trigger handover processes for changing from one AP to another.

If users of Mercury are only interested on optimizing the computation offloading side of the Fog Computing infrastructure, they can use a less detailed model that removes all the elements focused on connectivity, significantly simplifying the model under study and boosting up simulation performance for exploring a greater number of scenarios in less time. Figure 6 shows an schematic of the simplified Mercury model.

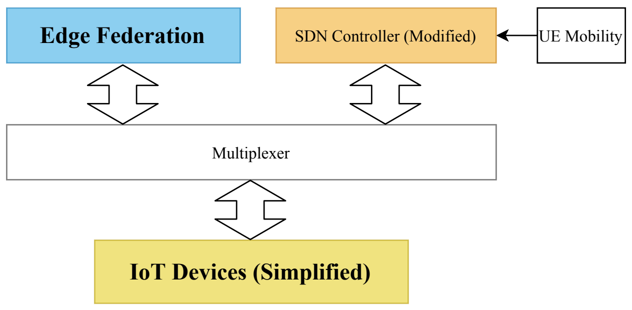


Figure 6: Mercury model with simplified connectivity.

In contrast with the bypassed version of Mercury, the simplification is not structural, but behavioral (i.e., some of the components of the model behave in a different way). While validating the behavior of the base model validates as well the bypassed version, an additional study must be provided to prove that the simplified version is valid as well.

The edge federation layer remains intact, as it has nothing to do with end users connectivity. However, UE is simplified, removing any submodule in charge of connecting to the RAN: now, UE is always connected to the SDNC itself. Depending on the location of UE and the status of the edge federation, the SDNC redirects computation offloading to the most suitable EDC. The AMF, Access Points, Crosshaul, and Radio layers are removed in this model, as they just model network connectivity-related issues.

In this model, UE does not require any AP to connect to the network. Therefore, the AP discovery, RAN access, and handover processes disappear. This change can also affect to the delay perceived by end users. However, depending on the motivation of the user of Mercury, these differences may not be significant enough. In these cases, the simplified model of Mercury provides experiment results in significantly less time than the bypassed and original versions, as proved in Section 4. Table 1 compares the multi-faceted models implemented in Mercury and the output provided by them.

Table 1: Models capabilities comparison.

Model	Original	Bypassed	Simplified
EDC Resource Management	✓	✓	✓
Edge Federation Resource Management	✓	✓	✓
UE Mobility	✓	✓	✓
RAN Connectivity	✓	✓	✗
Physical Networking	✓	✗	✗

4 PERFORMANCE EVALUATION

In this section, we perform a set of simulations using all the multi-faceted models defined in Mercury to compare them, and discuss the differences in the outputs provided by each of them. Finally, we study the simulation speedup achieved by each alternative model.

4.1 Scenario Description

The scenario under study consists of the incremental learning Advanced Driver Assistance System (ADAS) use case presented in our previous work (Cárdenas et al. 2020). UE devices are vehicles executing a Machine Learning (ML) model for detecting potential hazardous situations. Data stream is sent to the nearest available EDC. The computation offloading application consists of an incremental learning service for each vehicle. To do so, EDCs inject in real-time the new images to the training process of the predictive models. Once they significantly outperform the onboard versions, the embarked models are upgraded on the fly. The scenario, shown in Figure 7, is based on real mobility traces of taxis in the San Francisco bay area (Piorkowski, Sarafijanovoc-Djukic, and Grossglauser 2009). EDCs of the edge federation are represented as squares. APs are depicted as starts. The color of each AP coincides with the color of their closest EDC. Light blue dots correspond to UE mobility traces.

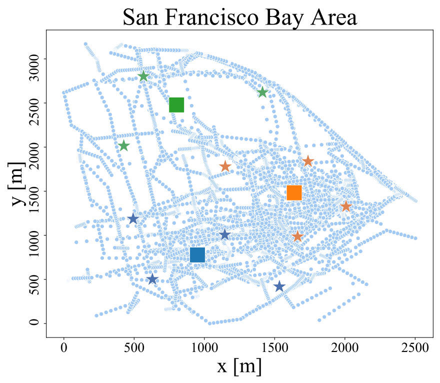


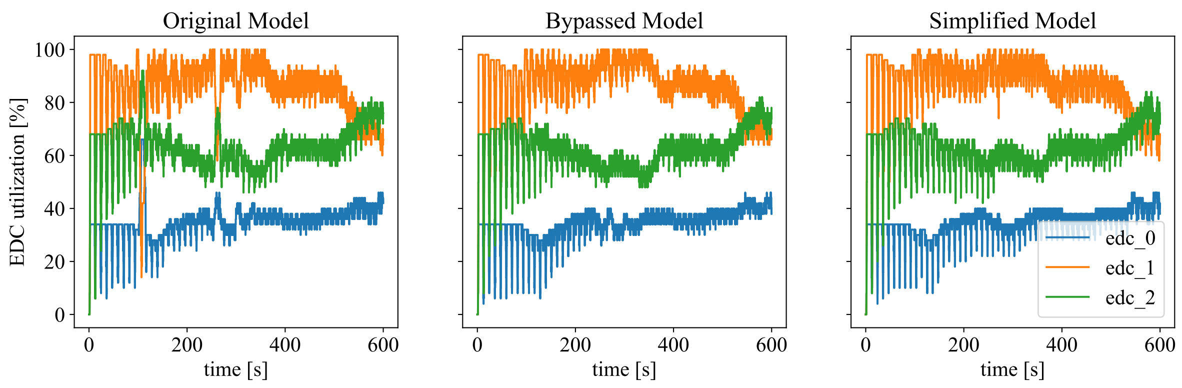
Figure 7: Scenario under study.

For each alternative model of Mercury (i.e., original, bypassed, and simplified), we simulated 10 minutes with a variable number of UE devices in the scenario, ranging from 10 to 100. The output provided by each simplified model regarding the utilization of EDCs computing resources was compared with the original model to infer the error caused by the simplifications of the overall scenario. Finally, from the time required for simulating each scenario, we analyze the speedup achieved by each model simplification, and provide a regression curve that estimates the time required for simulating 10 minutes as a function of the number of UE devices in the scenario. Simulations were run on a MacBook Pro Retina, 15-inch Mid 2015, 2.5 Quad-Core Intel Core i7 with 16 GB 1600 MHz DDR3 memory, using the PyCharm 2020.1.1 IDE. All simulation were executed in sequential mode.

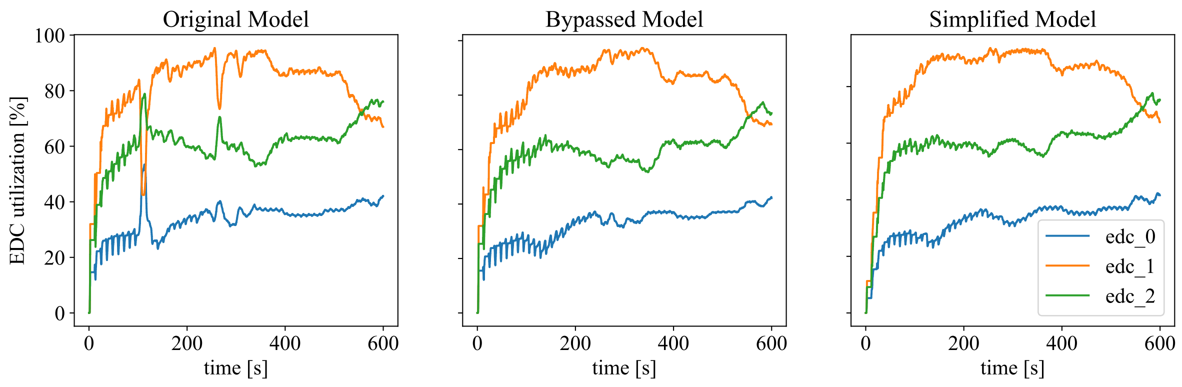
4.2 Simulation Results

In every use case scenario explored in this research work, the three alternative models of Mercury provided a similar output regarding the resource management of the EDCs. Figure 8 shows the output data of all the different models for the scenario that contained 100 UE devices (i.e., the most crowded and complex). As shown in Figure 8(c), the resource utilization trend line for every EDC coincided. However, the original model of Mercury registered two anomalies at $t \approx 140$ s and $t \approx 250$ s that were not reported in the output of the bypassed nor the simplified models.

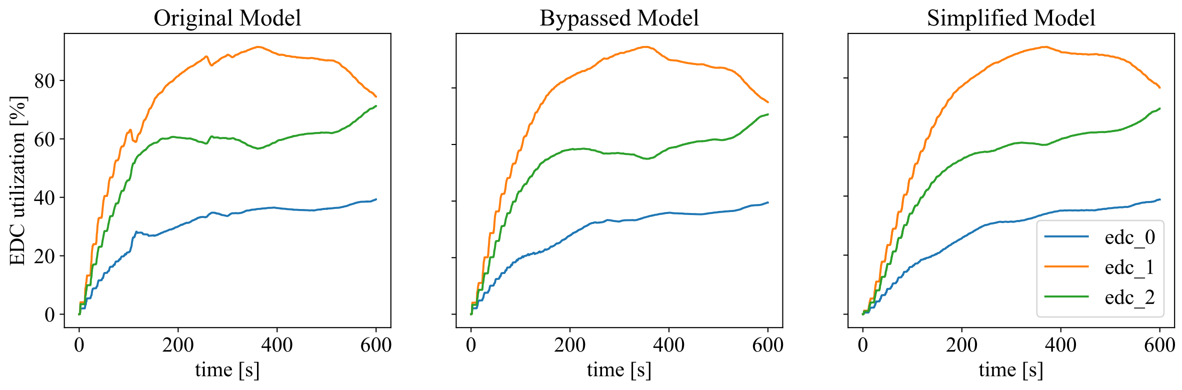
Analyzing the output provided by the original model, both anomalies happen when the EDC 1 (in orange) reaches 100% of its resource utilization. As this EDC has no more available resources for new incoming computation offloading session requests, the SDNC removes this EDC from the availability list, routing potential new sessions to the other EDCs that, though further from the end users, have enough available resources for hosting new computation offloading services. Once the EDC 1 closes ongoing sessions, the SDNC adds it again to the availability list, and communicates to the nearest APs that they must re-route new service requests to this EDC again. However, in the original model, due to the communication delays imposed by the physical layer of the crosshaul network, this message is not received instantaneously by the APs and, in the mean time, new sessions are still routed to the alternative EDCs (this explains the



(a) Original output.



(b) Output filtered with Exponential Moving Average ($\alpha = 0.01$).



(c) Output filtered with Exponential Moving Average ($\alpha = 0.001$).

Figure 8: Edge Data Centers resource utilization for 100 UEs.

simultaneous increase on the resource utilization of EDCs 0 (in blue) and 2 (in green)). Sessions remain open until UE requests to close them. In this scenario, the session time was set to 20 seconds. After 20 seconds, UE closes their computation offloading session, idle for 1 second, and proceed to open a session again. When this new session is requested, APs already have their routing tables updated, and route them again to the EDC 1. On the other hand, as there is no communication delay in the bypassed nor the simplified models, the leakage of UE when an EDC is congested to other EDCs of the edge federation is less pronounced.

Figure 9 shows the time required for simulating the scenarios on each model implemented by Mercury. If we

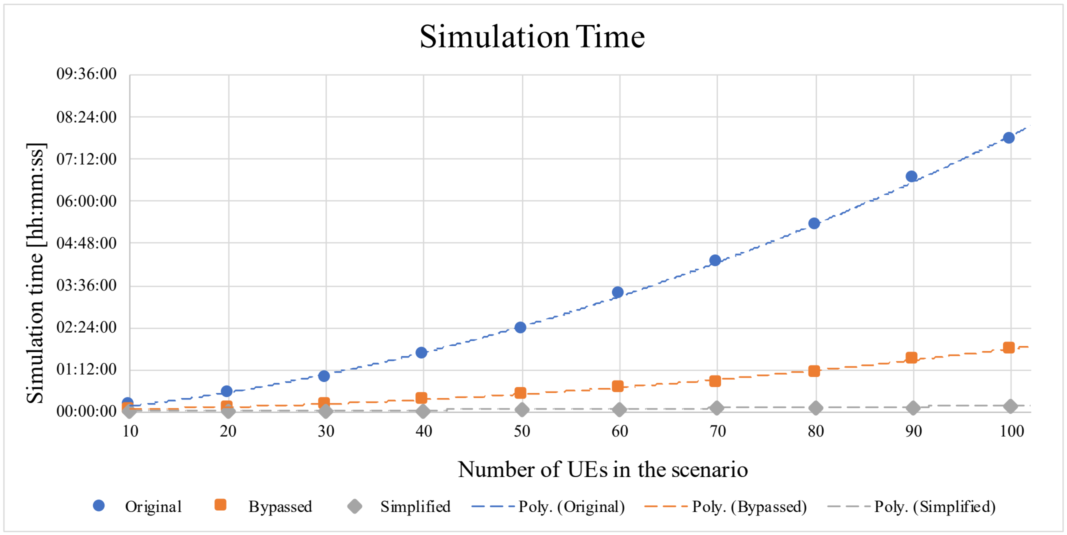


Figure 9: Simulation time for different models and scenarios.

consider that the complexity of every model is $O(n^2)$, where n is the number of UE devices in the scenario, the time required for simulating 10 minutes of the scenario can be deduced from the following equations:

$$S_{\text{original}}(n) = 2.09 \cdot n^2 + 78.96 \cdot n - 404.6 \tag{4}$$

$$S_{\text{bypassed}}(n) = 0.62 \cdot n^2 + 0.93 \cdot n + 233.75 \tag{5}$$

$$S_{\text{simplified}}(n) = 0.03 \cdot n^2 + 3.67 \cdot n + 17.73 \tag{6}$$

These trend lines are depicted in Figure 9 as dashed lines. While the original model required 7 hours and 47 minutes to simulate the scenario with 100 UE devices, the bypassed model took only 1 hour and 49 minutes, whereas the simplified model finished in 10 minutes and 39 seconds. Table 2 shows the speedup of the models compared with each other, computed as the ratio between the simulation times.

The speedup obtained with the bypassed model compared to the original seems to be constant around 4 (i.e., the bypassed model introduces four times less simulation overhead than the original model). Indeed, looking at Equations (4) and (5), we see that the coefficient C_2 of the polynomial is approximately 4 times greater in the original model than in the bypassed model. On the other hand, the simplified model increases its speedup as the number of UE devices rises, going from 14.51 for 10 devices to 43.84 for 100 devices. Analyzing Equation (6), we see that the second-degree coefficient is significantly lower than in Equation (4). We deduce from this that the degree two of the polynomial that describes the complexity of the simplified model is not predominant for 100 UE devices or less. If we assume that the complexity of the simplified model is $O(n^2)$ too, the maximum speedup of the simplified model would be $\lim_{n\to\infty} \frac{S_{\text{original}}(n)}{S_{\text{simplified}}(n)} = 69.66$.

Implementing a multi-faceted model for Mercury, we achieved to reduce simulation times of a complex system from several hours to minutes. This improvement enables us to explore a higher number of scenarios

Table 2: Simulation speedup of models defined by Mercury.

UE	Original VS Bypassed	Bypassed VS Simplified	Original VS Simplified
10	3.59	4.04	14.51
20	4.06	4.98	20.24
30	4.06	5.55	22.53
40	4.27	7.10	30.31
50	4.68	6.93	32.39
60	4.77	7.65	36.51
70	4.91	7.90	38.77
80	4.68	8.86	41.41
90	4.42	9.95	44.00
100	4.29	10.23	43.84

in less time, optimizing the resource management mechanisms of edge computing infrastructures while reducing the overall development time. Furthermore, as we keep a stack of models with incremental degree of detail in our M&S&O framework, we can always validate the outcome against highly detailed, richer models, ensuring full coverage of the solution and reducing potential future risks and failures.

5 CONCLUSIONS

M&S&O tools have become a common practice for developing complex systems in recent years, as they enable to improve the performance and robustness of the application under study. However, enforcing full coverage when validating these tools is still a significant challenge, implying unaffordable simulation overheads. In this context, we present a multi-faceted model description for Mercury, an M&S&O framework for IoT applications that make use of Fog Computing infrastructures for computation offloading tasks. We defined two alternative models that, while keeping the structure of the original one, removed complexity from elements that affected less to the resource management of Fog Computing infrastructures. The output provided by these models was very similar to the original one under a realistic ADAS use case. However, we obtained these results up to 44 times faster than using the initial model. With this new proposal, researchers would be able to optimize resource allocation algorithms faster and then validate the optimized elements with less effort against a more realistic model that ensures full coverage of the solution. As future work, we will use the multi-faceted model of Mercury to explore more complex resource management algorithms without facing time-consuming simulations, contributing in that way to the definition of more efficient computing infrastructures for enabling novel IoT applications.

ACKNOWLEDGMENT

This work has been partially supported by the Education and Research Council of the Community of Madrid (Spain) under research grant P2018/TCS-4423, the Centre for the Development of Industrial Technology (CDTI) under contract IDI-20171194, and the Spanish Ministry of Science and Innovation under contract RTC-2017-6090-3.

REFERENCES

3GPP 2019. “Release 15”. Technical report, 3rd Generation Partnership Project.

Cárdenas, R., P. Arroba, R. Blanco, P. Malagón, J. L. Risco-Martín, and J. M. Moya. 2020. “Mercury: A modeling, simulation, and optimization framework for data stream-oriented IoT applications”. *Simulation Modelling Practice and Theory* vol. 101, pp. 102037. Modeling and Simulation of Fog Computing.

Dalle, O., B. P. Zeigler, and G. A. Wainer. 2008. "Extending DEVS to support multiple occurrence in component-based simulation". In *2008 Winter Simulation Conference*, pp. 933–941. SCS.

de Sousa Junior, W. T., J. A. B. Montevechi, R. de Carvalho Miranda, and A. T. Campos. 2019. "Discrete simulation-based optimization methods for industrial engineering problems: A systematic literature review". *Computers & Industrial Engineering* vol. 128, pp. 526 – 540.

Gepperth, A., and B. Hammer. 2016. "Incremental learning algorithms and applications". In *European Symposium on Artificial Neural Networks (ESANN)*. Bruges, Belgium.

Gupta, H., A. Vahid Dastjerdi, S. K. Ghosh, and R. Buyya. 2017. "iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments". *Software: Practice and Experience* vol. 47 (9), pp. 1275–1296.

Karbalaei, A., D. Turgut, M. Dagley, E. Vasquez, and H. J. Cho. 2018. "Collaborative Multidisciplinary Engineering Design Experiences in IoT (Internet of Things) for Teachers Through Summer Research Site Program". In *ASME 2018 International Mechanical Engineering Congress and Exposition*. American Society of Mechanical Engineers Digital Collection.

Mittal, S., and A. Tolk. 2020. *Complexity Challenges in Cyber Physical Systems: Using Modeling and Simulation (M&S) to Support Intelligence, Adaptation and Autonomy*. John Wiley & Sons.

Piorkowski, M., N. Sarafijanovoc-Djukic, and M. Grossglauser. 2009, January. "A Parsimonious Model of Mobile Partitioned Networks with Clustering". In *The First International Conference on COMmunication Systems and NETworkS (COMSNETS)*. COMSNETS Association.

Russell, M. 2012. "Using MBSE to enhance system design decision making". *Procedia Computer Science* vol. 8, pp. 188–193.

Shannon, C. E. 1948. "A Mathematical Theory of Communication". *Bell System Technical Journal* vol. 27.

Shaw, J. A. 2013. "Radiometry and the Friis transmission equation". *American journal of physics* vol. 81.

Sonmez, C., A. Ozgovde, and C. Ersoy. 2018. "EdgeCloudSim: An environment for performance evaluation of edge computing systems". *Trans. on Emerging Telecommunications Technologies* vol. 29 (11).

Yuan, Q., H. Zhou, J. Li, Z. Liu, F. Yang, and X. S. Shen. 2018, Jan. "Toward Efficient Content Delivery for Automated Driving Services: An Edge Computing Solution". *IEEE Network* vol. 32 (1), pp. 80–86.

AUTHOR BIOGRAPHIES

ROMÁN CÁRDENAS received the MSc degree in Telecommunication Engineering in 2019 from Technical University of Madrid (UPM), where he is currently pursuing the PhD in Electronic Systems Engineering. His research interests include modeling and simulation with applications in the IoT domain. He can be reached at r.cardenas@upm.es.

PATRICIA ARROBA is an Assistant Professor at the Technical University of Madrid (UPM). She received her PhD degree in Telecommunication Engineering from UPM in 2017. Her research interests include energy and thermal-aware modeling and optimization of data centers. Her email address is p.arroba@upm.es.

JOSÉ M. MOYA is an Associate Professor in Technical University of Madrid (UPM). He received his PhD degree in Telecommunication Engineering from UPM in 2003. His research interests include proactive and reactive thermal-aware optimization of data centers. His email address is jm.moya@upm.es.

JOSÉ L. RISCO-MARTÍN received his PhD from Complutense University of Madrid, and currently is Associate Professor in the Department of Computer Architecture and Automation at Complutense University of Madrid. His research interests include computer-aided design and modeling, simulation and optimization of complex systems. He can be reached at jlrisco@ucm.es.