Performance optimization of edge computing homeland security support applications

Marco Gribaudo
Dipartimento di Elettronica, Informatica e Bioingegneria
Politecnico di Milano
via Ponzio 51
20133, Milano, Italy

Mauro Iacono
Dipartimento di Matematica e Fisica
Università degli Studi della Campania "L. Vanvitelli"
viale Lincoln 5
81100 Caserta, Italy

Agnieszka Jakóbik
Institute of Computer Science
Cracow University of Technology
Warszawska st 24,
31-155 Cracow, Poland

Joanna Kołodziej
Research and Academic Computer Network (NASK)
Kolska st 12, 01-045
Warsaw, Poland

KEYWORDS

Edge computing; Cloud computing; Performance evaluation; Emergency response; Queuing networks; Genetic algorithms.

ABSTRACT

Critical distributed applications have strict requirements over performance parameters, that may affect life of users. This is a limitation that may prevent the exploitation of cost effective solutions such as Cloud Computing (CC) based architectures: in fact, the quality of the connection with the CC facility and the lack of control on cloud resources may limit the overall performances of an application and may cause outages. A way to overcome the problem, and disclose the advantages of CC to critical applications, is provided by Edge Computing (EC). EC adds local support to CC, allowing a better distribution of application tasks according to their timeliness requirements. In this paper we present an innovative Special Weapons And Tactics (SWAT) support application, designed to empower effective operations in wide scenarios, that leverages EC to join CC elasticity and local immediateness, and we exploit Queuing Networks (QN) and Genetic Algorithms (GA) to design and optimize the system parameters for an effective workload distribution.

I. INTRODUCTION

Critical systems are systems characterized by strict requirements (in general, on dependability, performances, or also security attributes), to avoid consequences that may be dramatic in terms of economic loss or threats to human life. In order to satisfy such requirements, the design process of the system should carefully be planned, and the use of models to support the design of performance parameters during the life cycle of the system may provide a significant help in design choices.

In this paper, the focus is on performance modeling of a reference architecture for a novel application, oriented to support complex SWAT operations in high risk scenarios involving hostages and criminals or terrorists. The system, inspired to a firefighting support system presented for the first time in [4], assists a number of SWAT teams on the field by providing sensors, augmented reality tools, tactical information and life protection support to enhance the effectiveness of the team while raising the level of protection for agents, by a real time analysis of the conditions of the field and the possibility of computationally complex coordination and data management applications to lower the impact of possible menaces (e.g., terrorists hiding between hostages or their organized reaction to the SWAT action).

In order to provide a flexible, yet cost effective solution, that allows to combine highly available and timely tasks with heavy and variable workloads, to support both field oriented, tactical and strategic operations, the application is based on EC. This poses the problem of having an optimal dimensioning of the various components of the architecture according to the needs of each mission, or to the evolution of the scenario. This requires the model to be optimized to find the right balance between the different tasks on the system, and the evaluation and optimization have to be feasible with a low computational effort and in a short time, in order to keep the system well tuned along all the duration of the mission.

The EC architecture has been chosen because of its high flexibility [12]. In the EC paradigm CC facilities are complemented with additional computing facilities located where the application has to be used. These additional facilities are "at the edge of the cloud" (hence the name), so that they are close to the user and do not suffer from the problems that affect the communications between the user and the CC facility. The edge facilities may be common computers, mobile devices, low power/low cost devices (e.g. Raspberry Pi), or other devices.

As performances are a critical issue since the first phases of the design cycle and along the life cycle of the system, and the system is subject to be deployed in different alternative configurations in relation to scenarios, we complemented a performance evaluation technique, based on QN, with a parameter optimization technique, based on GA. The choice of QN is due to

Proceedings 32nd European Conference on Modelling and Simulation ©ECMS Lars Nolle, Alexandra Burger, Christoph Tholen, Jens Werner, Jens Wellhausen (Editors)
ISBN: 978-0-9932440-6-3/ ISBN: 978-0-9932440-7-0 (CD)

the fact that QN are a consistent, yet simple way to consider all relevant aspect of the system, easily understandable for non specialists and, in general, widely supported by available tools and easily included into more complex modeling frameworks [2][1].

The choice of GA is due to their generality and the fast and reliable evaluation of the possible parameter configurations. They enable to find the near optimal solution of the NP-hard optimization problems faster then traditional 'hard' computing methods, [16], [17].

In this paper, we present a performance evaluation approach for a complex, scalable, critical EC based SWAT operation support and management system, designed to assist missions involving one or more SWAT teams in high-risk scenarios, that leverages on wearable and deployable sensors, augmented reality devices and CC support for strategical mission supervision and additional support, to allow an effective coordination of SWAT teams, providing life parameters monitoring, extended environmental modeling and real time information enhancement.

The paper is organized as follows: after this introduction, Section II introduces the system, III describes the modeling approach, IV presents performance evaluation. Conclusions follow.

II. A CRITICAL EDGE COMPUTING BASED SYSTEM

The proposed system is inspired to the one presented for the first time in [4] and described in details in [**?**]. In these papers a different QN performance model has been presented, together with some performance related considerations, but no optimization support.

The architecture of the system leverages on the commercial availability of several key technologies, namely *Wireless Sensors Networks* (WSN) (including optical and thermal cameras) and related *Internet of Things* (IoT) and *Mobile Sensors Networks* (MSN) [9][15], *Augmented Reality* (AR), *Wearable Computing* (WC), and the already introduced EC and CC. These technologies have been considered, for their maturity level and the existence of cost effective devices on the COTS market, to satisfy the 4 main functionalities that allow to accomplish the mission of the system: *field control*, *agent support*, *tactical support* and *strategical support*. For a matter of clarity, let us consider an example scenario: a terrorist attack into a skyscraper.

A. System functionalities

Field control consists in the capability of providing real time information about what is happening on the field. In our example scenario, the SWAT teams may deploy wireless and/or mobile sensors and cameras to take control and tactical superiority in the building, and may wear sensors to monitor the environment; additionally, a (partial) supplemental view on the environment may be obtained by means of drones, either moving inside or inside the building, so to obtain a partition of the overall scenario into an observable and a non observable part, in which the observable part is directly monitored by the SWAT team leader. Additionally, the building may have an existing sensor network or cameras, eventually partially damaged by the attackers, that may be somehow remotely accessible by the mission leader, outside the action scope of the SWAT team leader.

Agent support is provided by the equipment that is given to each agent. For the purposes of this paper, this equipment consists of wearable sensors to monitor the environment and the health conditions of each agent, and AR support to provide real time additional data superposed to the physical perception of the environment, eventually integrated with a WC device to preprocess data. Data are exchanged in real time with the SWAT team leader, that reports to the mission leader, by means of the tactical support.

Tactical support consists in allowing each SWAT team leader to coordinate his team, control the observable part of the scenario and to take decisions harvesting the information about the current tactical situation on the field. The SWAT team leader should be able to communicate with his team and to control mobile sensors, including drones and cameras or other complex devices. Tactical support is provided by a real time information system running on a dedicated server, locally hosted on the field (e.g. into a van) and capable of performing real time analysis on gathered sensor data and providing updated AR information to the agents belonging to a SWAT team. With reference to the example scenario, one or more SWAT teams are on the field, each one supported by a van in which the SWAT team leader operates, located in close proximity with respect to the skyscraper so to avoid communication problems with the SWAT team.

Strategical support consists in providing the mission leader, which operates remotely, with updated information about the SWAT teams on the field and the field itself, by merging sensed data and external data sources, and additional advanced data analysis tools (e.g. face recognition applications) to obtain strategical superiority and to evaluate the possible alternative scenarios resulting from possible course of events and the consequences of the various options that the SWAT team leader may decide to examine before taking his decisions. In the example, the commander may use a mission supervision environment leveraging information from the various SWAT teams and from external sources such as the internal sensing system of the skyscraper, accessed by the World Wide Web, and Homeland Security department (or equivalent governmental or non-governmental institutions) databases to simulate the course of events or wider scenarios, and the effects of different actions of the SWAT teams. Processing is not real time, but may require elastic computing resources and interfacing with other local or national authorities, such as police, air force, secret services and the aforementioned Homeland Security authorities.

B. System architecture

In order to implement the 4 functionalities of the system, the architecture is organized into 4 main subsystems, namely *Cloud Backend* (CB), *Edge Frontend(s)* (EF), *Personal Support* (PS), *Sensing Network* (SN): they are meant to provide the implementation platform respectively for the strategical support, the tactical support, the cyberfireman support and the field control. A representation of a possible configuration is in Fig. 1.

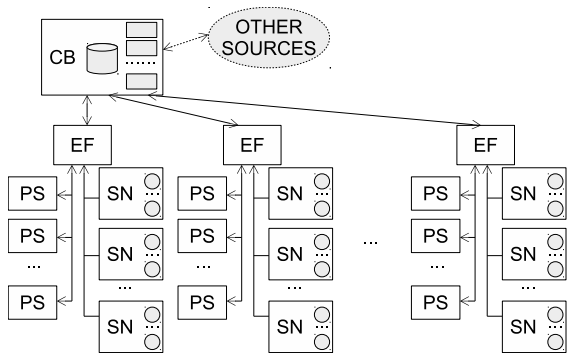

Fig. 1. System architecture layout

CB provides the needed elasticity for the execution of the needed strategical superiority and mission supervision applications, that may vary from a simple decision support system to complementary applications such as signal processing, OLAP, image recognition, simulation, integrating field sensed data and external sources accessed on line, obtained from other authorities in real time and/or other databases already available in the system or special purpose applications. In our example, the CB may run advanced signal processing applications to find signals into the data complex provided by the wearable sensors that agents cannot perceive, such as the voices of nearby hostages or of terrorists, not directly audible by the agents, or can perform data fusion between data from SN and PS and from the building sensor network to provide references about the actual viability of the skyscraper or the position of people by thermal sensing, and integrate AR data or lower the complexity of real time generation of AR data for the EF. Consequently, workload may significantly vary according to the phases of the mission and different needs, that may in principle be not defined before they actually manifest. An important parameter is the volume of service requests coming from the EFs and the amount of supervisory requests generated on the CB by the mission leader (and, possibly, other authorities).

EF provides computing resources for the real time execution of the SWAT team coordination tasks. It is basically provided by a high performance server (e.g. as in [5]), that is connected to the CB by a high speed mobile connection (e.g. 4G or 5G) and is installed on a van. In the following, we will not consider the connection problems with PS and SN that may arise because of the difficult conditions, as it is out of the scope of this paper: we hypothesize, without loss of generality, that the quality of the connection is kept with the use of proper repeaters (e.g. by setting up an ad-hoc network). The EF workload is supposed to be regular and dependent on the number of agents in a SWAT team and on the number and type of sensors in the controlled SN.

PS provides an agent (similarly to what presented in [8]) what is needed to implement wearable sensing and AR by means of a WC system. A minor computing effort is performed locally, basically devoted to service tasks and devices management, and communications with the related EF. Data from sensors (e.g. position, health monitoring, environmental sensors and additional deployable ones), with partial preprocessing, are sent to EF, and commands and AR information are received from the EF. Workload is mainly related to the configuration of the equipment, and sensors contribute to the workload of the EF.

SN consists in a traditional sensor network, in which sensors may be fixed, mobile, disposable, actuable, multimedia or complex sensors. All sensors are controlled by the EF, and affect its workload. The kind and number of sensors in the network may vary, even during a same mission, because of damages, of new needs, of energy exhaustion, of replacement (e.g. see [9][15]).

According to the scenario, the actual configuration may vary, and proper criteria for a correct parameter setting is crucial in order to ensure that performances are sufficient to match the requirements. A configuration will be composed of one CB, one EF per each SWAT team operating in the field, one PS per each agent and one or more SN for each EF.

III. PERFORMANCE MODEL

We model system with the multi-class open queueing network shown in Figure 2. Queuing stations are used to model EF nodes (`Edge Frontend`), network communication (`Edge Network`) and CC nodes (`Cloud Backend`). Arrivals to the `Edge Frontend` nodes represent data acquired from the field. Since different data, of different complexity, can be collected at different speeds, we use costumer classes to model traffic heterogeneity. Without loss of generality, in this work we focus on two different classes of data: the one generated by SN devices, and the one originating from PS nodes. We call λ_{SN} and λ_{PS} respectively the data rate of the two sources. Since we consider scenarios where more EF are present, we model each one with a different queuing station. We call N_{EF} the number of frontend nodes, and we suppose that each of them is characterized by the number of devices it receives from / sends to: for this reason we might have different arrivals for each EF station. For sake of simplicity, we consider EF nodes to be homogeneous, all characterized by the same arrival rate from same classes, and each node providing the same performance and thus serving its requests in the same amount of time. However, the service time for different traffic classes might differ: in particular we call $S_{EF,SN}$ and $S_{EF,PS}$ the average service time required by the two considered data

acquisition classes. To simplify the evaluation of the system, we suppose that EF nodes are equipped by an operating system that, when more requests needs to be served, processes them concurrently. In this way EF nodes can be modeled adopting the processor sharing service discipline.

All nodes send data to the cloud using the same network connection, hence communication is modeled with a single queuing station to which all the EF nodes are connected. Service times might be different for the considered sensor classes, and in particular we set them to $S_{N,SN}$ and $S_{N,PS}$. The main feature of edge computing is that data collected by the sensors is first analyzed by the EF nodes, and only a fraction of them is required to be sent through the network. We model this by allowing jobs of the two classes to immediately leave the EF stations respectively with probabilities $1-p_{SN}$ and $1-p_{PS}$. In this way only a fraction p_{SN} and p_{PS} of data produced by the sensors reaches the cloud.

The cloud part of the application is considered to distributed between N_C equivalent virtual machines that share the arriving load in an uniform way. Besides data received from the EF nodes, the cloud backend is queried also by other users that monitor the event from different institutions. This is modeled by adding an extra class, characterized by a global arrival rate λ_C. Traffic generated by this particular class immediately leaves the system after being served. Data incoming from the sensors (SN and PS classes) go instead back to the EF nodes to allow feedback from the central infrastructure to the server deployed on the field. We call respectively $S_{C,SN}$, $S_{C,PS}$, $S_{C,C}$ the average service time required by the three considered classes. Table III summarizes model parameters and shows the base-line value considered in the following part of the paper. Values have been validated as realistic from experts of the field.

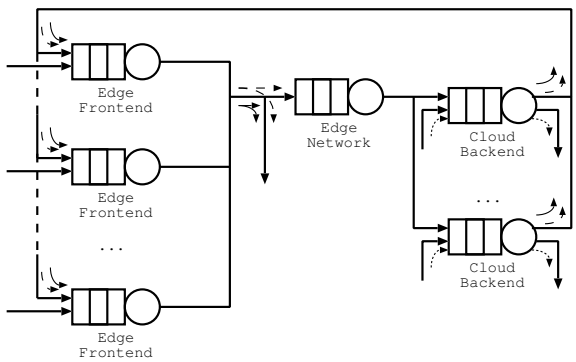

Fig. 2. Queuing network model for the system

IV. SCENARIOS AND ANALYSIS

The model presented in Figure 2 can be used to study several deployment scenarios. We will focus on a couple of them: maximizing the system computational power, while minimizing the cost for acquiring resources; and considering the impact of the coordination activity on the cloud on the system response time.

Param.	Description	Value
N_{EF}	Num. of EF nodes	2
λ_{SN} λ_{PS}	Arrival rates of SN and PS classes	1.25 data/s. 1.25 data/s.
$S_{EF,SN}$ $S_{EF,PS}$	Service times at the EF nodes	0.2 s. 0.35 s.
p_{SN} p_{PS}	Probability of going to the cloud	0.18 0.12
$S_{N,SN}$ $S_{N,PS}$	Network transfer times	10 ms. 15 ms.
N_C	Number of VMs	2
λ_C	Coordination traffic	0.1 req./s.
$S_{C,SN}$ $S_{C,PS}$ $S_{C,C}$	Service times at the cloud backend	1.5 s. 1 s. 2 s.

TABLE I: The model parameters

A. MAXIMIZING SYSTEM POWER

As defined by Kleinrock in [19], *system power* corresponds to $\Pi = \frac{X}{R}$: the ratio of the throughput and the response time of a system, or of one of its components. The main rationale about this performance index is that it increases when either the throughput increases, or when the response time decreases. In other words, the configuration in which a component works at its maximum system power corresponds to the point where users receives the best tradeoff between speed and latency. In this first study we want to determine the point that maximizes system power, considering a fixed total data generation rate. In particular, we imagine that data is collected at a total rate of Λ readings per second, which is a structural parameter that depends on the specific sensing technology and communication protocol between the sensors and the EF that is employed. The choice is then to properly balance reading from the two class of sensors. In particular, we call $0 \leq \beta \leq 1$ the *traffic mix* parameter, and we define $\lambda_{SN} = \beta\Lambda$ and $\lambda_{PS} = (1-\beta)\Lambda$. Another parameter we want to take into consideration is the number of EF nodes that is being used. Indeed, the higher the number of EF, the higher will be the system power of the application. However, considering a larger number of EF nodes implies a higher cost: in particular we consider a discount function $g_\mu(N_{EF})$, that describes the impact of having a higher cost due to a larger number of EF nodes, on the average system power we aim to maximize. In this example we set $g_\mu(N_{EF}) = \frac{\mu}{N_{EF}}$, with $\mu > 0$. The rationale is that a smaller number of nodes has an higher impact on the system power we aim to maximize. We thus create an objective function $f_1(\beta, N_{EF})$ defined as:

$$f_1(\beta, N_{EF}) = \alpha \frac{\lambda_{SN}}{R_{SN}(\beta, N_{EF})} + (1-\alpha)\frac{\lambda_{PS}}{R_{PS}(\beta, N_{EF})} + \frac{\mu}{N_{EF}} \quad (1)$$

where $R_{SN}(\beta, N_{EF})$ and $R_{PS}(\beta, N_{EF})$ represent the response time of the two sensor data classes. Constant α is used instead to balance the interest between the sensors ($\alpha = 1$) and the personal devices ($\lambda = 0$).
Figure 3 shows the system power of the two data classes and the value of the objective function 1 for different traffic mixes β and for a different number of nodes N_{EF}. As it can be seen, the system power Π_{SN} of the sensor network has a monotonic behaviour, while the one of the personal support class Π_{PS} shows a maximum when the number of EF nodes $N_{EF} = 1$. The figure also shows the values of the objective function $f_1(\beta, N_{EF})$ for $\alpha = 0.5$, and $\mu = 0.5$ or $\mu = 1$, to consider two different cost scenario where data classes are equally important. As it can be seen, in some cases behaviour is monotonic, while in some other it experiences some internal convexity. This motivates the use of optimization algorithms to study the values of β and N_{EF} where this maximum is reached.

Figure 4 shows the different values of the parameter

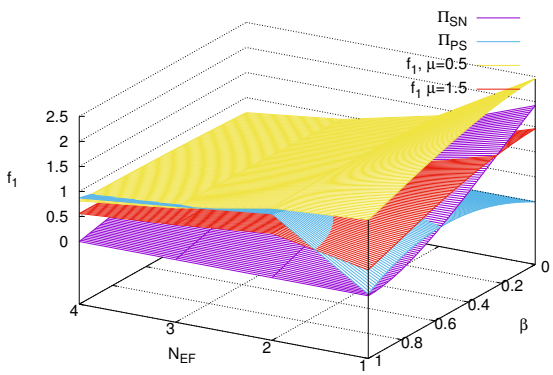

Fig. 3. Ibjective function $f_1(\beta, N_{EF})$ for different values of resource cost parameter μ and $\alpha = 0.5$.

that maximizes the objective function for different cost parameters μ, and the balance parameters α. When the cost is very high, it is better to keep as few EF nodes as possible: however, depending on the balance parameter α there could be optimal non-trivial traffic mixes that optimizes the system. When the cost is very low, it is not always true that it is better to use a larger number of EF nodes: the balance parameter α can make the system work at its optimum configuration even with $N_{EF} = 1$, and $N_{EF} > 2$ are never needed.

B. THE EFFECT OF THE COORDINATORS

Different players in the management of the event monitor the evolution of the scenario accessing the cloud back-end of the application. Depending on their update rate λ_C, they might have a clearer and more up-to-date picture of the scenario, increasing their ability to help and their effectiveness in solving the situation. However, a larger upload rate might overload the system, leading to an unstable behaviour. In this study, we consider the effect on the response time of the traffic generated by the coordinators, varying $0.1 \leq \lambda_C \leq 0.5$

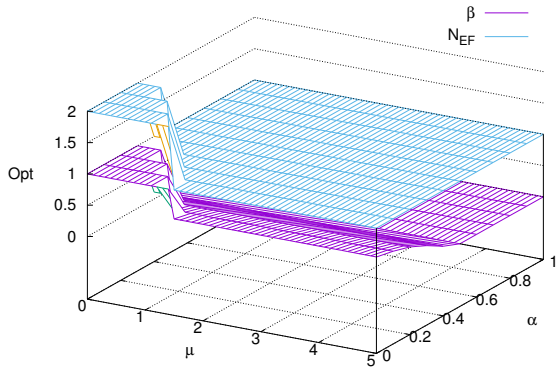

Fig. 4. Optimum values of β and N_{EF} as function of the cost parameter μ, and the balance parameter α.

req./s., and we combine the analysis with the traffic mix introduced in the previous section. We measure the non-effectiveness of the coordination actions with a function $l_\theta(\lambda_C) > 0$: larger values denotes a less effective coordination of forces, while values that tends to 0 denotes optima interactions. In this work we use $l_\theta(\lambda_C) = \frac{\theta}{\lambda_C}$, where $\theta > 0$ is a scaling parameter to make the measure compatible with the expected response times. Again we want to account the cost on the infrastructure in our performance assessment. In this case, however, since response time must be minimized, it must increase with the number of resources. Moreover, since we are studying the coordinators, which mainly affect the cloud, we must account not only for the number of EF nodes N_{EF}, but also for the number of provisioned VMs N_C. In particular, we use a cost function $h_\nu(N_{EF}, N_C)$, and we set it to $h_\nu(N_{EF}, N_C) = \nu \cdot (N_{EF} + N_C)$ (with $\nu > 0$ as a metric conversion parameter), and we set the objective function $f_2(\beta, N_{EF}, N_C, \lambda_C)$ to:

$$\begin{aligned} f_2(\beta, N_{EF}, N_C, \lambda_C) \quad = \quad & \frac{\lambda_{SN}}{\Lambda^*} R_{SN}(\beta, N_{EF}, N_C) + \\ & \frac{\lambda_{PS}}{\Lambda^*} R_{PS}(\beta, N_{EF}, N_C) + \\ & \frac{\lambda_C}{\Lambda^*} R_C(\beta, N_{EF}, N_C) + \\ & \frac{\theta}{\lambda_C} + \nu \cdot (N_{EF} + N_C) \end{aligned} \quad (2)$$

where $R_C(\beta, N_{EF}, N_C)$ is the response time of the coordinator class, and $\Lambda^* = \Lambda + \lambda_C$ is the total system throughput.
Figure 5 shows the value of the objective function $f_2(\beta, N_{EF}, N_C, \lambda_C)$ for different traffic mix β and coordinator traffic λ_c in different resource configuration. As it can be seen, there are many cases in which the system is not stable (represented in the figure with $f_2(\beta, N_{EF}, N_C, \lambda_C) = 0$), and the objective function is concave in some cases. Figure 6 shows for which values of β and λ_C the minimum of the objective function is reached. It is interesting to see that, when the cost

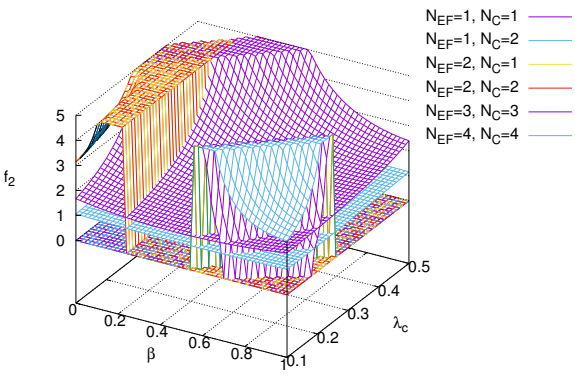

Fig. 5. Objective function $f_2(\beta, N_{EF}, N_C, \lambda_C)$ for different combinations of EF nodes N_{EF} and cloud VMs N_C.

is very high; the minimum is obtained for $\beta = 1$ and $\lambda_C = 0.18$ req./s. When the cost is lower, the minimum is obtained in different combinations of β and λ_C. The optimization algorithm also determined the optimum number of nodes N_{EF} and N_C: even if not explicitly shown in the figure to make it more readable, the minimum of the objective function is reached when the minimum number of resource is used when $\beta < 1$. In the flat area with $\beta = 1$ instead, the optimum is reached when the maximum number of resources is used.

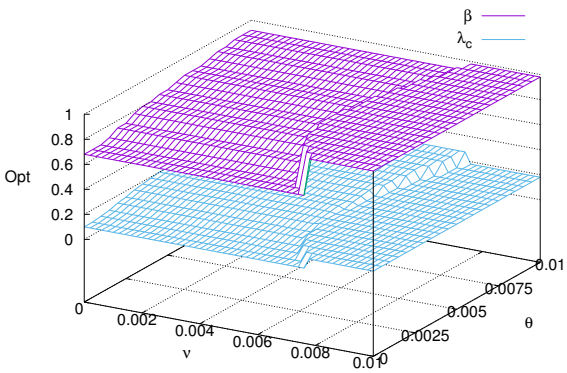

Fig. 6. Optimum values of β and λ_C as function of the cost parameter ν, and coordination effectiveness θ.

V. RELATED WORKS

The integration of Cloud computing, mobile computing and Internet of Things (IoT) allows to exploit a greater flexibility in performance engineering and tuning with respect to cloud architectures, and extends their great available power and elasticity by allowing an efficient collection and transmission of massive and ubiquitous data sources, such as IoT devices and mobile devices, to overcome the limitations of the weak point of cloud architectures, the network connection towards the edge of the cloud. The readers may refer to [23] for a first approach to edge architectures problems and features, while the general context of application of these solution is presented in [12] and [21], that report about security related and law related topics. Of course, a correct balance, and related tuning and adaptation, of system parameters is crucial to achieve desired overall performance figures, as discussed in [22], [24], [26], [20] and [6], that may help readers in understanding the general framework in which responsiveness, scalability, privacy and fault tolerance in edge architectures may be achieved and modeled or measured. Due to the interest about edge-based solutions, there is an intense research activity that spans over performances and system/software management, communication protocols and architectural solutions. In [10] the fog computing perspective and some related applications are presented. In [18] the role of containers in edge-based software architectures is discussed and quantitatively documented, and in [14] the general software architectural aspects are examined. In [7] the focus is on the relation of edge architectures with IoT, with special attention to communication problems. In [11] an autonomic approach to edge architecture management is discussed. In [25] the edge approach is extended towards osmotic computing. For the aspects focusing specifically on cloud related problems and performance modeling, readers may refer to other previous papers of the authors, stating their positions that influenced this work ([16], [17], [3], [13]. In the first two papers authors successfully applied a specifically designed GA for solving NP-hard task scheduling problems for Computational Clouds. Their novel approach assumed generating every consecutive population from 'freezing' the worst parents instead of removing them from the population. New population was of the equal size as the previous population. This enabled to generate more broad sub-optimal solutions space and hence find the solution faster.

VI. CONCLUSIONS AND FUTURE WORK

Results show the effectiveness of the presented modeling approach for the performance evaluation of critical edge based systems, that showed a non trivial behavior. Future work include a better exploitation of GA in the technique and a more detailed characterization of the system.

VII. ACKNOWLEDGEMENTS

This article is partially based upon work from COST Action IC1406 High-Performance Modelling and Simulation for Big Data Applications (cHiPSet), supported by COST (European Cooperation in Science and Technology)

REFERENCES

[1] S. Balsamo and A. Marin. Queueing networks. In M. Bernardo and J. Hillston, editors, *Formal Methods for Performance Evaluation, 7th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM 2007, Bertinoro, Italy, May 28-June 2, 2007, Advanced Lectures*, volume 4486 of *Lecture Notes in Computer Science*, pages 34–82. Springer, 2007.

[2] E. Barbierato, M. Gribaudo, and M. Iacono. Exploiting multiformalism models for testing and performance evaluation in SIMTHESys. 2011.

[3] E. Barbierato, M. Gribaudo, and M. Iacono. Modeling and evaluating the effects of Big Data storage resource allocation in global scale cloud architectures. *International Journal of Data Warehousing and Mining*, 12(2):1–20, 2016.

[4] E. Cavalieri d'Oro, S. Colombo, M. Gribaudo, M. Iacono, D. Manca, and P. Piazzolla. Modeling and evaluating performances of complex edge computing based systems: a fire-fighting support system case study. In *Valuetools 2017,*, 2018.

[5] D. Cerotti, M. Gribaudo, M. Iacono, and P. Piazzolla. Modeling and analysis of performances for concurrent multi-thread applications on multicore and graphics processing unit systems. *Concurrency and Computation: Practice and Experience*, 28(2):438–452, 2016. cpe.3504.

[6] H. Chang, A. Hari, S. Mukherjee, and T. V. Lakshman. Bringing the cloud to the edge. In *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 346–351, April 2014.

[7] M. Chiang and T. Zhang. Fog and iot: An overview of research opportunities. *IEEE Internet of Things Journal*, 3(6):854–864, Dec 2016.

[8] S. Colombo, M. Gribaudo, M. Iacono, D. Manca, and P. Piazzolla. A low-cost distributed IoT-based augmented reality interactive simulator for team training. In *Proceedings of the 31th European Conference of Modelling and Simulation, May 23-26 2017, Budapest*, page to appear, 2017.

[9] M. D'Arienzo, M. Iacono, S. Marrone, and R. Nardone. Estimation of the energy consumption of mobile sensors in wsn environmental monitoring applications. In *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*, pages 1588–1593, March 2013.

[10] A. V. Dastjerdi and R. Buyya. Fog computing: Helping the internet of things realize its potential. *Computer*, 49(8):112–116, Aug 2016.

[11] M. Desertot, C. Escoffier, and D. Donsez. Towards an autonomic approach for edge computing: Research articles. *Concurr. Comput. : Pract. Exper.*, 19(14):1901–1916, Sept. 2007.

[12] C. Esposito, A. Castiglione, F. Pop, and K. K. R. Choo. Challenges of connecting edge and cloud computing: A security and forensic perspective. *IEEE Cloud Computing*, 4(2):13–17, March 2017.

[13] M. Gribaudo, M. Iacono, and D. Manini. Improving reliability and performances in large scale distributed applications with erasure codes and replication. *Future Generation Computer Systems*, 56:773 – 782, 2016.

[14] Z. Hao, E. Novak, S. Yi, and Q. Li. Challenges and software architecture for fog computing. *IEEE Internet Computing*, 21(2):44–53, Mar. 2017.

[15] M. Iacono, E. Romano, and S. Marrone. Adaptive monitoring of marine disasters with intelligent mobile sensor networks. In *Environmental Energy and Structural Monitoring Systems (EESMS), 2010 IEEE Workshop on*, pages 38–45, 2010.

[16] A. Jakbik, D. Grzonka, and F. Palmieri. Non-deterministic security driven meta scheduler for distributed cloud organizations. *Simulation Modelling Practice and Theory*, 76:67 – 81, 2017. High-Performance Modelling and Simulation for Big Data Applications.

[17] A. Jakóbik, D. Grzonka, J. Kołodziej, and H. González-Vélez. Towards secure non-deterministic meta-scheduling for clouds. In *30th European Conference on Modelling and Simulation, ECMS 2016, Regensburg, Germany, May 31 - June 3, 2016, Proceedings.*, pages 596–602, 2016.

[18] M. B. A. Karim, B. I. Ismail, W. M. Tat, E. M. Goortani, S. Setapa, J. Y. Luke, and H. Ong. Extending cloud resources to the edge: Possible scenarios, challenges, and experiments. In *2016 International Conference on Cloud Computing Research and Innovations (ICCCRI)*, pages 78–85, May 2016.

[19] L. Kleinrock. Power and deterministic rules of thumb for probabilistic problems in computer communications. In *ICC '79; International Conference on Communications, Volume 3*, volume 3, pages 43.1.1–43.1.10, 1979.

[20] H. D. Park, O.-G. Min, and Y.-J. Lee. Scalable architecture for an automated surveillance system using edge computing. *J. Supercomput.*, 73(3):926–939, Mar. 2017.

[21] R. Roman, J. Lopez, and M. Mambo. Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges. *Future Generation Computer Systems*, 2016.

[22] M. Satyanarayanan. The emergence of edge computing. *Computer*, 50(1):30–39, Jan 2017.

[23] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646, Oct 2016.

[24] W. Shi and S. Dustdar. The promise of edge computing. *Computer*, 49(5):78–81, May 2016.

[25] M. Villari, M. Fazio, S. Dustdar, O. Rana, and R. Ranjan. Osmotic computing: A new paradigm for edge/cloud integration. *IEEE Cloud Computing*, 3(6):76–83, Nov 2016.

[26] S. Yi, Z. Hao, Z. Qin, and Q. Li. Fog computing: Platform and applications. In *2015 Third IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb)*, pages 73–78, Nov 2015.

AUTHOR BIOGRAPHIES

MARCO GRIBAUDO is an associate professor at the Politecnico di Milano, Italy. He works in the performance evaluation group. His current research interests are multi-formalism modeling, queueing networks, mean-field analysis and spatial models. The main applications to which the previous methodologies are applied comes from cloud computing, multi-core architectures and wireless sensor networks.

MAURO IACONO is a tenured assistant Professor and Senior Researcher in Computing Systems at Dipartimento di Matematica e Fisica, Università degli Studi della Campania "L. Vanvitelli", Caserta, Italy. His research activity is mainly centered on the field of performance modeling of complex computer-based systems, with a special attention for multiformalism modeling techniques. More information is available at http://www.mauroiacono.com.

AGNIESZKA JAKÓBIK (KROK) received her M.Sc. in the field of stochastic processes at the Jagiellonian University, Cracow, Poland and Ph.D. degree in the field of neural networks at Tadeusz Kosciuszko Cracow University of Technology, Poland, in 2003 and 2007, respectively. From 2009 she is an Assistant Professor. Her e-mail address is: agneskrok@gmail.com

JOANNA KOŁODZIEJ is an associate professor in Research and Academic Computer Network (NASK) Institute and Department of Computer Science of Cracow University of Technology. She is a Head of the Department for Sciences and Development. She serves also as the President of the Polish Chapter of IEEE Computational Intelligence Society. She is also a Honorary Chair of the HiPMoS track of ECMS. Her e-mail address is: joanna.kolodziej68@gmail.com. The detailed information is available at www.joannakolodziej.org