Proactive vs Reactive Failure Recovery Assessment in Combined Fog-to-Cloud (F2C) Systems

Vitor Barbosa Souza*§, Xavi Masip-Bruin§, Eva Marín-Tordera§, Wilson Ramírez§, Sergio Sánchez-López§
* Informatics Department (DPI), Universidade Federal de Viçosa (UFV), Brazil
§ Advanced Network Architectures Lab (CRAAX), Universitat Politècnica de Catalunya (UPC), Spain
vitorbs@dpi.ufv.br, {xmasip, eva, wramirez, sergio}@ac.upc.edu

***Abstract*—The increasing number of end user devices at the edge of the network, along with their ever increasing computing capacity, as well as the advances in Data Center technologies, paved the way for the generation of Internet of Things (IoT). Several IoT services have been deployed leveraging Cloud Computing and, more recently, Fog Computing. In order to enable efficient control of cloud and fog premises, Fog-to-Cloud (F2C) has been recently proposed as a distributed architecture for coordinated management of both fog and cloud resources. Certainly, many challenges remain unsolved in combined Fog-to-Cloud systems, mostly driven by the dynamicity and volatility imposed by edge devices, such as the recovery of failures at the edge of the network. Indeed, possible failures in computing commodities may be prohibitive for the achievement of the envisioned performance in F2C systems. In this work, we assess proactive and reactive strategies for failure recovery of network elements by modelling them as a Multidimensional Knapsack Problem (MKP) and study the impact of each one on several aspects such as service allocation time, recovery delay and computing resources load. The obtained results show the effect each strategy brings, thus concluding with some analysis on the recovery strategy best suiting distinct IoT scenarios.**

Keywords— Cloud computing, Fog computing, combined Fog-to-Cloud, Internet of Things, Failure recovery

I. INTRODUCTION

The Internet of Things (IoT) envisions a massive, heterogeneous set of devices demanding connectivity anywhere, anyhow, by any means, which has coined new concepts for the Future Internet such as Smart Home or Smart Cities [1], to name a few. These concepts leverage advances in Data Center (DC) technologies as well as in the Cloud Computing paradigm, which has, with no doubt been positioned as the key enabler for IoT applications development. This assessment is rooted on the fact that the huge computational capacity offered by cloud premises enables the deployment of IoT services with high requirements, including Video-on-demand or DC backup solutions.

More recently, the high end-to-end latency observed in cloud applications—due to the large distance separating end-user devices and cloud—has driven the need for a new paradigm, so-called fog computing [2], particularly addressed to support the highly demanding requirements of real-time IoT services, such as Smart Transportation or e-Health. In short, Fog Computing inherits the main concepts of Cloud Computing, but move them to the edge of the network. The main idea is to bring remote processing closer to end-user devices, enforcing locality, imposing low response time, low network load and less energy consumption. Since fog computing is not competing but complementing cloud computing, the next evolution in the cloud arena will be built upon considering both cloud and fog resources on a coordinated way to efficiently execute a wide range of services.

However, such a coordinated scenario, requires new management policies in place responsible for controlling and managing the large set of heterogeneous and volatile resources brough by combining fog and cloud. To that end a combined Fog-to-Cloud (F2C) management architecture has been recently proposed in [3]. F2C is intended to provide a coordinated fog and cloud resources allocation, thus enabling the distributed execution of IoT services in fog, cloud, or both, while simultaneously leveraging the heterogeneity perceived on the combination of Fog and Cloud Computing systems. As shown in Fig. 1, distinct resources in F2C are hierarchically organized into layers enabling IoT services to make the most out of the heterogeneous and distributed resources.

Albeit works such as [4] have assessed service allocation in F2C, several open issues are not addressed yet. For instance, the high dynamicity perceived at edge devices requires novel mechanisms for failure recovery, specifically designed for distributed scenarios. In fact, although distinct strategies have been successfully employed in Cloud Computing for protection of data and network resources, this is yet an open challenge in Fog computing, further inherited by F2C systems. It must be remarked that a key design aspect of such protection strategy is to enable service protection with low service allocation time and protection cost, i.e., the amount of computing resources devoted for service protection as well as the recovery delay.

In this paper, we introduce, model and analyze the service protection in F2C, taking into account two failure recovery strategies: i) the so-called proactive protection, where protection resources (or secondary) are pre-allocated and used in case of primary resource failure; and ii) the

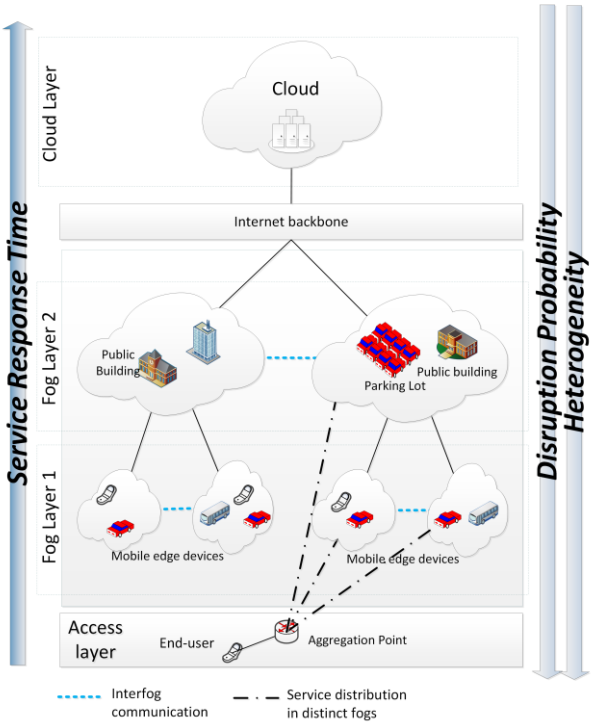


Fig. 1. Fog-to-Cloud (F2C) architecture.

reactive protection, where the secondary resources are not allocated until a failure occurs.

The rest of this paper is organized as follows. Section II introduces the related work in recovery strategies. Section III describes in a comprehensive manner the F2C architectural model. Afterwards, Section IV introduces the recovery strategies used in this work as well as presents the failure recovery problem formulation. Then, Section V provides numerical results regarding the implemented strategies. Finally, Section VI provides the final conclusions.

II. Related Work

In this section, we highlight the main research studies dealing with resilience in Cloud and Fog Computing, as well as existing contributions dealing with proactive and reactive recovery strategies. Both, proactive and reactive recovery techniques, are commonly used in distinct scenarios, including those making use of plenty of reliable computing and network resources, such as data centers [5], and those with unreliable and reduced amount of resources, such as wireless sensor networks [6]. The work presented in [7] utilizes a proactive strategy to recover from network failures by defining backup paths with distinct priorities in network switches, not requiring the controller participation on the recovery process. On the fog computing side, authors in [8] replicate the service execution between the edge device and a remote server. Albeit protection is not the main focus of that contribution, the proposed proactive strategy yields a fault-tolerant service execution.

Among the studies dealing with reactive protection in Cloud scenarios, the work available in [9] may be mentioned from a networking perspective. In that work Cloud resilience is achieved by means of network virtualization. Other works, such as [10], consider a Cloud scenario based on an optical network core, where an analogous technique to shared-protection, so-called shared-path shared-computing (SPSC) protection is used against a single link or node failure. Regarding resilience in Fog scenarios, the work in [11] addresses recovering from Mobile Edge Computing (MEC) failures. The strategy presented by the authors is based on workload offloading to neighboring resources aiming at the recovery for overloaded or broken MEC. Nevertheless, the availability of neighboring resources are not guaranteed by a protection strategy.

Despite the wealth of available studies related to Cloud computing resilience, to the best of our knowledge, there are still several open challenges dealing with the negative effects stemmed from the lack of Fog commodities availability. In an attempt to address these challenges, in this paper, we formalize, by means of linear programming, proactive and reactive protection strategies against commodity failures. Our goal is to provide enough computing resources to withstand single commodities failures. To that end, we assume a F2C scenario where only failures at the Fog infrastructure are possible. The rationale driving this assumption is to show the vulnerability of Fog nodes and its impact on service transmission performance and protection cost.

III. The Combined F2C Ecosystem

In the F2C architecture considered in this work, F2C resources are split into three hierarchical layers, consisting in one cloud layer and two fog layers. In a bottom-up perspective, the first layer is the fog layer 1, or Fog-1, which comprises edge devices located at a 1-hop distance from the end-user. Therefore, it is mostly composed of resource-constrained devices embracing the ones presenting high dynamicity, such as moving vehicles, wearables or smartphones, as well as static devices, such as sensors. The second layer is the fog layer 2, or Fog-2, which embraces edge devices presenting lower mobility, higher resource availability and higher distance from the end-user, if compared to the ones in fog layer 1. Vehicles in a parking lot or fog premises deployed in public buildings are examples of Fog-2 resources. Finally, the third layer is the cloud layer, or simply cloud, handled by traditional cloud service providers relying on data centers premises, with high computing capacity and availability as a tradeoff with the high end-user communication latency.

The total capacity of each fog and cloud as well as the capacity demanded by the services are measured in terms of slot units. Therefore, an slot is the minimum unit for resource allocation, i.e., the amount of slots required by one single service is static and their allocation may be done either in one single F2C node—i.e., fog or cloud—, offering the required amount of slots, or in a distributed fashion, where service slots may be allocated in distinct F2C nodes and even in distinct F2C layers. In addition, for the sake of simplicity, we consider the allocation of just one resource

type, i.e., we abstract the type of resource slot being requested by services and offered by F2C nodes. This assumption aims at the simplification of the problem modeling, allowing this work to focus on the description and comparison of the protection strategies.

Finally, also for the sake of simplicity, we consider the failure of one single fog node, that is to say, only one fog may become inaccessible on the first or second fog layers. Moreover, we assume that cloud service providers can withstand with internal failures through their own protection schemes, i.e., we consider that the cloud resources are always available.

IV. Failure Recovery in F2C

In this section, we introduce the failure recovery strategies assessed in this work, presenting main particularities as well as a formal description.

A. Failure Recovery Strategies

In this subsection, we go deep into the proactive and the reactive failure recovery strategies modeled in this paper. It is worth mentioning that, in both strategies, the allocation of protection resources is done in a horizontal fashion. This means that the protection is allocated at the same fog layer in the architecture, i.e., the protection resources for a fog in Fog-1 are located in a fog also at Fog-1, and so on. Horizontal allocation may ease the allocation of resources with similar characteristics, respecting SLA even after failure events. The assessed strategies work as follows.

- Proactive recovery: in this strategy, protection resources are pre-allocated for each primary resource allocation. If the primary resource becomes unavailable, the protection may be used with no extra allocation delay.
- Reactive recovery: in this strategy, the protection resources are not allocated until failure occurrence. Indeed, a set of protection resources may be selected to react to a failure, but they are not allocated until the failure takes place. This strategy allows a set of protection resources to be shared among primary resources allocated in distinct fogs, diminishing the resource underutilization whilst delaying the protection allocation.

B. Problem Model

In this section, we provide a formal description of the protection strategies assessed in this work by modeling the failure recovery problem as a Multidimensional Knapsack Problem (MKP) whose objective is twofold: 1) decrease the delay for transmission of each service; and 2) decrease the protection cost –by reducing the slots consumed for protection–, and diminish the recovery latency –by provisioning low-delay protection slots using resources located in lower F2C layers. Therefore, the objective function, as described in (1), minimizes the sum of delays for the allocation of each service in the set S (first problem objective) and the sum of the cost for accessing each protection slot in the F2C resources in the set R (second problem objective). All symbols used in the presented model are defined on Table I.

$$\text{Min:} \quad \sum_{i \in S} D_i + \sum_{r \in R} P_r \tag{1}$$

In order to represent the primary and secondary slot allocation into the available F2C resources, the respective linear programming integer variables Y and X were defined as follows.

$$Y_{i,r,k} = \begin{cases} 1, & \text{if service } i \text{ is allocated in resource } r \\ & \quad \text{consuming slot } k \\ 0, & \text{otherwise} \end{cases}$$

$$X_{r,k} = \begin{cases} 1, & \text{if resource } r \text{ has its slot } k \text{ reserved as} \\ & \quad \text{a secondary slot (protection)} \\ 0, & \text{otherwise} \end{cases}$$

TABLE I. Model Symbols Definition

Symbol	Definition
D_i	Delay for the transmission of the service *i* (all primary slots required by the service)
P_r	Recovery delay offered by resource *r*
S	Set of services to be executed
U_i	Total number of slots required to execute service *i*
R	Set of F2C resources, i.e., set of distinct cloud and fog nodes
K_r	Set of slots provided by F2C resource *r* for both primary and secondary use
L_n	Set of fogs available at fog layer *n*
F_r	Set of fog nodes in the same fog layer of fog *r*
T_r	Allocation delay of a slot in F2C resource *r*, according to its F2C layer

Moreover, the objective function is subject to a set of constraints, as described in the following lines.

$$\sum_{r \in R} \sum_{k \in K_r} Y_{i,r,k} * T_r = D_i \,, \forall i \in S \tag{2}$$

$$\sum_{k \in K_r} X_{r,k} * T_r = P_r \,, \ \forall r \in R \tag{3}$$

$$\sum_{r \in R_s} \sum_{k \in K_r} Y_{i,r,k} = U_i \,, \ \forall i \in S \tag{4}$$

$$\sum_{i \in S} \sum_{k \in K_r} Y_{i,r,k} + \sum_{k \in K_r} X_{r,k} \leq |K_r| \,, \ \forall r \in R \tag{5}$$

$$\sum_{i \in S} Y_{i,r,\mathrm{k}} + X_{r,\mathrm{k}} \leq 1 \,, \ \forall r \in R \ \wedge \ \forall k \in K_r \tag{6}$$

The sum of the allocation delay of each service (D_i) is given by (2), whilst the sum of the recovery delay introduced by the transmission latency on each node P_r, is defined by (3). The overall protection cost of each node is defined as the number of slots consumed for protection multiplied by the delay of a single transmission to this node. Notice that the protection cost in the model may be reduced by diminishing the amount of protection slots allocation X or by using resources with lower transmission delay T. On the other hand, the primary allocation delay can only be reduced through the selection of resources offering lower delay T, since the amount of primary slots allocation cannot be decreased. This constraint is defined by (4).

Jointly with (4), equations (5) and (6) are responsible for the coordinated allocation of primary and secondary slots

according to the service requirements and the resources availability. Therefore, constraint (4) aims at ensuring the allocation of the required amount of primary slots for each service in S. In addition, this constraint enables the distributed service allocation into distinct F2C resources and layers, as envisioned by F2C. The constraint imposed by (5) guarantees that each resource capacity, in term of slots, is not surpassed. This is achieved by considering the capacity of each F2C node and its allocation of both primary slots demanded by distinct services and secondary slots required by protection strategies. Furthermore, (6) is responsible for constraining the capacity of each slot to, at maximum, one allocation. That is to say, it ensures that each consumed slot is allocated as either one primary or one secondary slot.

In order to cope with the distinct protection strategies analyzed in this work, two strategy-specific constraints are introduced. Equation (7) ensures the protection allocation for the proactive strategy, whilst (8) guarantees the reactive recovery strategy. The horizontal strategy defined for the proposed recovery mechanisms is also ensured by both equations. Therefore, on each primary slot allocated in one layer, (7) enforces the reservation of a protection slot in the same layer. On the other hand, the goal of (8) is accomplished by granting the number of protection slots on each fog layer do not be lower than the amount of primary slots allocated in any individual fog located in the same layer.

$$\sum_{i \in S} \sum_{r \in L_n} \sum_{k \in K_r} Y_{i,r,k} = \sum_{q \in L_n} \sum_{k \in K_q} X_{q,k}\,, \quad \forall n \in \{1,2\} \qquad (7)$$

$$\sum_{i \in S} \sum_{k \in K_r} Y_{i,r,k} \leq \sum_{q \in F_r - \{r\}} \sum_{k \in K_q} X_{q,k}\,, \quad \forall r \in L_1 + L_2 \qquad (8)$$

V. Preliminary Results

In this section, we evaluate the impact of the presented protection strategies on the overall service transmission delay, failure recovery delay, and F2C resources load. To that end we use PuLP [12] and Gurobi Optimizer [13]. The simulation results were obtained taking into account the service parameters and resource parameters shown, respectively, in Table II and Table III.

TABLE II. Simulation parameters: Services

Parameter	Value
Number of requested services	From 10 to 150
Ratio between amount of service types: low consuming / high consuming services	90 (low) / 10 (high)
Slots demanded by low consuming services	2
Slots demanded by high consuming services	20

TABLE III. Simulation parameters: F2C Resources

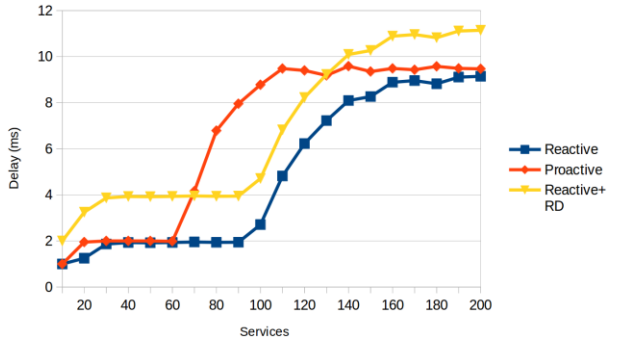
Parameter	Fog layer 1	Fog layer 2	Cloud
Number of F2C nodes per layer	8 fogs	4 fogs	1 cloud
Available resources per node	10	100	Unlimited
Allocation delay per F2C layer	1 ms	2 ms	10 ms

In this work, we assumed that, for each service, the resource allocation in distinct F2C nodes is performed in a parallel fashion. Therefore, the overall time demanded for the allocation of each service corresponds to the time for allocation of the resource with the highest delay.

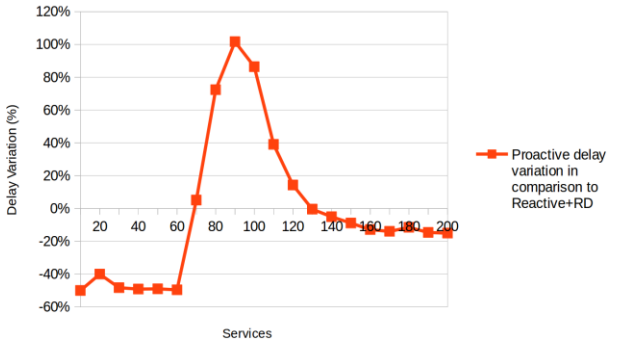
The impact in terms of delay perceived on both failure recovery strategies is illustrated by Fig. 2(a). Notice that this figure presents the delay for service transmission in reactive and proactive recovery, evidencing the lower delay for the former. Nevertheless, this single results confrontation is unfair since proactive schemes consume more resources in lower layers for protection purposes leading to a premature primary allocation in higher layers, with higher latency. To cope with this situation, we also calculate the overall delay for reactive strategies considering both the service transmission to the primary resource and delay added by the service allocation after one eventual fail. Albeit the delay for the service request transmission in reactive strategies cannot be overlooked, proactive strategies offer seamless recovery, hence recovery latency is not applicable, as previously exposed in this paper. Also shown in Fig. 2(a), the so-called reactive+RD consists in the overall delay for reactive strategy including the fail recovery delay. Therefore, a more realistic comparison, confronting the delays for proactive and reactive+RD results, reveals that, in failure scenarios, each strategy may present better performance in distinct cases according to the number or services to be executed. The fast delay increase observed in this figure can be explained by the shift on the service allocations from fog to cloud due the lack of available resources in lower layers. Consequently, even taking into account one eventual reactive recovery, the early depletion of fog layer resources, perceived in proactive strategies, results in a higher delay of this strategy under medium load. As can be observed in Fig. 2(b), a direct comparison between proactive and reactive+RD yields an increase from 50% lower to 100% higher delay of proactive recovery in comparison with the reactive+RD one, being followed by a decrease on the difference as the amount of services increase.

In order to analyze the resource utilization in the third F2C layer, Fig. 3 presents the allocation of primary slots in the cloud. For sake of comparison, we included also the results obtained by exclusive cloud allocation, i.e., the usage of a traditional cloud computing system with no fog layers, showing that the distributed allocation strategy envisioned by F2C enables an average load reduction in cloud resources as high as 50%. In this figure, we do not show the protection allocation in cloud commodities since we assume that the cloud servers implement their own protection techniques as previously pointed out in this paper. Moreover, it is worth mentioning that one of the benefits of the employed horizontal protection strategy is that the fog layers are not tied to the cloud regarding the protection allocation.

An analysis of the presented results shows that both proactive and reactive failure recovery are feasible on F2C architectures, albeit the higher underutilization of edge resources perceived in proactive strategies shall hinder the full employment of these strategies in scenarios with large amounts of services requests, such as Smart Transportation,



(a)



(b)

Fig. 2. Delay in evaluated strategies: (a) absolute and (b) relative comparison.

where reactive strategies may be preferred. Future lines of work include the study of the tradeoff between network and computing resources in proactive recovery strategies in F2C computing systems. Indeed, proactive strategies may enable a desirable redundancy on sensitive services requiring real-time fault-tolerant execution. Nevertheless, this redundancy may drain the already scarce energy resources in devices at the edge of the network. On the other hand, their allocation as primary and backup computing resources requires the continuous transmission of the service execution state in order to allow the backup execution to take place with minimum impairment in an eventual failure occurrence.

VI. Conclusion

The dynamicity of IoT scenarios imposes several challenges for the management of resources at the edge of the network and fault-tolerant service execution architectures such as F2C computing. In this work, we discuss two strategies for resource failure recovery. In order to evaluate the employment of the discussed strategies, we model the failure recovery problem as a MKP aiming at minimizing both the service transmission delay and the protection cost, taking into account the recovery latency and optimal protection resources distribution. The presented results showed that the employed strategy has a big impact on the service transmission and recovery performance.

Acknowledgments

This work was supported by the Spanish Ministry of Economy under contract TEC2015-66220-R, by the Catalan Government under contract 2014SGR371 and by the H2020 EU mF2C project. Vitor Barbosa Souza is supported by CAPES Foundation, Ministry of Education of Brazil, Proc. No.11888/13-0.

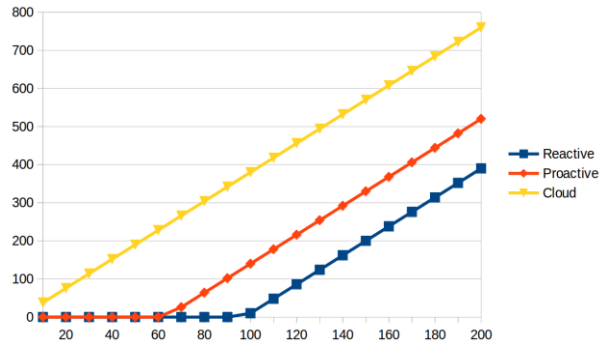


Fig. 3. Primary resource allocation in the Cloud.

References

[1] A. Zanella, N. Bui, A. Castellani, L. Vangelista, M. Zorzi. Internet of things for smart cities. In: IEEE Internet of Things journal, 1(1), pp. 22-32, February 2014.

[2] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog Computing and Its Role in the Internet of Things". In: Proceedings of 1st MCC Workshop on Mobile Cloud Computing, ser. MCC '12. New York, NY, USA: ACM, pp. 13–16, August 2012.

[3] X. Masip-Bruin, E. Marin-Tordera, A. Jukan, G. J. Ren, G. Tashakor. Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud (F2C) computing systems. In: IEEE Wireless Communications Magazine, 23(5), pp. 120-128, October 2016.

[4] V.B.Souza, X.Masip-Bruin, E.Marín-Tordera, W.Ramírez. Towards Distributed Service Allocation in Fog-to-Cloud (F2C) Scenarios. In: IEEE Global Communications Conference, Globecom 2016.

[5] I. F. Akyildiz, A. Lee, P. Wang, M. Luo, W. Chou. Research challenges for traffic engineering in software defined networks. In: IEEE Network, 30(3), pp. 52-58, 2016.

[6] M. Younis, I. F. Senturk, K. Akkaya, S. Lee, F. Senel. Topology management techniques for tolerating node failures in wireless sensor networks: A survey. Computer Networks, 58, pp. 254-283, 2014.

[7] A. Sgambelluri, A. Giorgetti, F. Cugini, F. Paolucci, P. Castoldi. OpenFlow-based segment protection in Ethernet networks. In: Journal of Optical Communications and Networking, 5(9), pp. 1066-1075, 2013.

[8] Y. W. Kwon, E. Tilevich. Energy-efficient and fault-tolerant distributed mobile execution. In: IEEE 32nd International Conference on Distributed Computing Systems (ICDCS), pp. 586-595, IEEE, 2012.

[9] I. B. B. Harter, D. A. Schupke, M. Hoffmann, G. Carle, Network virtualization for disaster resilience of cloud services. In: IEEE Communications Magazine, vol. 52, no. 12, pp. 88-95, December 2014. doi: 10.1109/MCOM.2014.6979957.

[10] C. Natalino et al., "Dimensioning optical clouds with shared-path shared-computing (SPSC) protection". In: 2015 IEEE 16th International Conference on High Performance Switching and Routing (HPSR), Budapest, 2015, pp. 1-6.

[11] D. Satria, D. Park, M. Jo, "Recovery for Overloaded Mobile Edge Computing", In: Future Generation Computer Systems, 2016.

[12] Optimization With PuLP. Available: http://www.coin-or.org/PuLP/

[13] Gurobi Optimization. [Online]. Avaliable: http://www.gurobi.com/