Budget and Performance-efficient Application Deployment along Edge-Fog-Cloud Ecosystem

Polona Štefanič
Cardiff University, School of
Computer Science & Informatics
Queen's Buildings, 5 The Parade,
CF243AA, UK
Email: StefanicP@cardiff.ac.uk

Omer F. Rana
Cardiff University, School of
Computer Science & Informatics
Queen's Buildings, 5 The Parade,
CF243AA, UK
Email: RanaOF@cardiff.ac.uk

Vlado Stankovski
University of Ljubljana,
Faculty of Civil and Geodetic Engineering
Jamova cesta 2,
1000 Ljubljana, Slovenia
Email: vlado.stankovski@fgg.uni-lj.si

***Abstract*—Applications that make use of Internet of Things (IoT) capture an enormous amount of raw data from sensors and actuators, which is frequently transmitted towards the cloud data centres for processing and analysis. However, due to varying and unpredictable data generation rates and network latency, sending the data towards a cloud data centre can lead to a performance bottleneck. With the emergence of Fog and Edge computing hosted microservices, data processing could be moved towards the network edge. We propose a novel Pareto-based approach that makes use of a multi-criteria bin packing optimisation for efficient and optimal distributed deployment of microservices – along edge, fog/cloudlet and cloud tiers. This optimisation takes account of non-functional requirements, such as operational cost, compute resource utilisation, service availability, response time, latency and similar. The results show that the present approach provides an optimal and sustainable consumption of compute resources and improves Quality of Service of the application during its runtime. The approach can also be integrated into software engineering workbenches for the creation and deployment of cloud-native applications, enabling partitioning of an application across the multiple infrastructure tiers outlined above.**

***Keywords*—Edge/Fog/Cloud computing, Quality of Service, IoT, Microservice, Pareto front**

I. Introduction

The number of Internet of Things (IoT) devices that change their physical location has grown significantly in the last few years. According to estimates by Forbes [1] and Cisco [2], by 2025 it is expected that such Internet-connected devices will reach the multi-billion mark, all generating data which has to be sent through the public network towards cloud data centres. Currently, IoT devices send data directly to cloud systems for processing and analysis to benefit from high availability, scalability, high power computing, unlimited storage and pooled computing resources on the pay-per-use and self-service models [3]. With the emergence of IoT and consequently increasing size of data, the reconfigurability of on-demand computing, such as Cloud computing, has become critical in order to support Quality of Service (QoS) of cloud-native applications.

Real time applications, in areas such as disaster warning systems, real time video analysis and deep learning models related to object, speech and text recognition have time varying demands on compute resources (based on the generation and availability of data). Sending and processing such data in a cloud data centre presents bottlenecks due to high latency, transmission costs, and privacy issues. Recently, with the emergence of Fog and Edge computing the trend has become to move the services, data and processing power towards the network edge for processing and analysis.

A novel approach is presented for budget- and performance-efficient deployment of a data pipeline based on microservices along Edge-Cloudlet/Fog-Cloud ecosystem. A traffic management use case is considered, which makes use of deep learning models for object recognition. The proposed Pareto-based multi-criteria bin packing optimisation approach considers a variety of non-functional requirements that can be divided into mandatory (hard constraints) and desirable (soft constraints), and returns optimal non-dominant solutions as a trade-off among conflicting objectives. These trade-offs include operational cost and network latency for efficient microservice deployment. Our key contributions are: (1) novel approach for supporting optimal and efficient distributed microservice placement across Edge-Fog/Cloudlet-Cloud ecosystem and (2) enhanced QoS of the overall cloud-native application based on data processing pipeline realised through microservices.

The rest of this paper is organised as follows: Section II discusses related work, followed by Section III where we introduce the motivating application scenario related to traffic management. Section IV proposes an Edge-Fog-Cloud architecture that is suitable for the distributed deployment of a data analysis pipeline. In Section V we formulate the problem as a multi-criteria bin packing optimisation approach. In Sections VI and VII we discuss experimental setup and provide results and evaluation of the approach. We reveal our future research directions and conclude the paper with Section IX.

II. Related Work

Recently, big data [4] and video analytics workflow pipelines [5] have become popular and widely used in business and scientific applications. A workflow pipeline in this instance is generally composed of multiple stages, where each stage represents a computational function/service. For returning more accurate and real-time results such services need to execute on high performance computing resources with low latency, and make effective use of geo-distributed data processing across multiple data centres [6].

Workflow stages can consist of microservices which are capable of being deployed within a Edge-Cloudlet/Fog-Cloud environment. Recently, a new paradigm called Osmotic Computing [7] was also proposed, providing an abstraction for the execution of lightweight microservices at the edge of the network coupled with more complex microservices running within a cloud data centres. Osmotic computing proposes mechanisms for migrating services between the edge and cloud systems based on performance and security "triggers", enabling an application to adapt its behaviour over time. Bonomi et al. [8] propose a hierarchical distributed architecture for the execution of applications along Edge-Fog nodes and cloud data centres.

Moreover, processing video streams in the clouds and at the edge of the network has also gained importance and popularity recently, due to possibility for use of increasing availability of elastic computing environments that support real-time resource allocation and provisioning [9]. For instance, Zamani et al. [10] propose a model for leveraging the use of computational resources towards the edge, fog and cloud data centres. They focus mostly on video stream analytics. Similarly, Ananthanarayanan et al. [11] argue that geographical distribution of cloud data centres and edge nodes closer to IoT devices is the only solution in order to meet strict real-time requirements of large-scale live video streams. Furthermore, Knight et al. [12] have investigated QoS metrics of time-critical CUDA applications, and provide a survey of key requirements for QoS-aware execution of such applications in cloud data centres.

III. Application scenarios

A traffic management scenario is used to illustrate the benefits of using both edge and cloud resources. A traffic management system needs to acquire data from fixed field sensors and autonomous vehicles in real time. A variety of fixed sensors are placed along roads and highways as road side units for detecting traffic conditions, such as traffic flow and congestion monitoring, vehicle, density and incident detection, overspeeding etc. Additionally, autonomous vehicles communicate with one another and with road side units and transmit their information on speed and location for forecasting potential congestion and to estimate travel time [13].

In this scenario, data sent from IoT devices to cloud-based systems must be processed and analysed in near real-time. Sending the data to the clouds means high latency and consequently longer response time to end users. On the other hand, processing data at the network edge and fog nodes offers low latency and consequently faster response to traffic events and therefore better application QoS.

The traffic management system implies the following components: (i) *Input sensors* are installed at fixed distance from one another on roads and collect and send data to a fog device for analysis; (ii) *Global Display Actuators* receive up-to-date responses from controllers or router fog devices and present parameter values such as lane closure signs or average vehicle speed, vehicle route displays and similar; (iii) *Controller Fog device* is responsible for compute, storage and network resources; it receives input from sensors and sends updates to the actuator; (iv) *Router Fog device* is a router linked to the controller fog device and enables multiple communication channels; moreover, it is also capable of support computational analysis; (v) *Cloud data center* is responsible for maintaining the entire application state. The workflow pipeline showing interaction between these different components is provided in Figure 1.

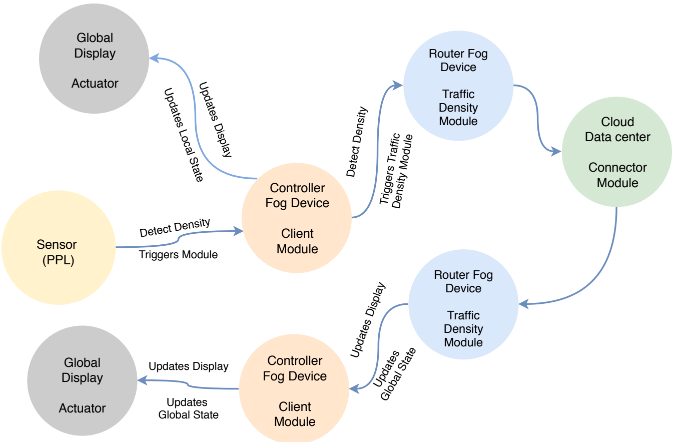

Fig. 1: Traffic Management scenario: Integrating Fog and Cloud.

IV. Edge-Fog-Cloud System Architecture

Our proposed system architecture consists of three tiers: edge, fog or cloudlet nodes and cloud data centres, as illustrated in Figure 2. At the edge of the network, fixed and dynamically positioned IoT devices generate data that is transmitted over a public network. The generated raw data in-transit, passes through fog or cloudlet nodes and continues to the cloud data centres. Fog nodes or cloudlets (to support mobility) are positioned between the network edge and the cloud.

Initial data pre-processing can be performed on the edge and fog nodes, however more complex tasks such as object detection and recognition (which can have greater demands on computing resources) should consequently be processed in cloud data centres. Therefore, we propose the distribution of the data pipeline functions, such as (i) data collection, cleaning and data pre-processing and, on the other hand, (ii) object detection and recognition through deep learning models to be executed across edge, fog and cloud infrastructure respectively as illustrated in Figure 2. We treat each functionality of the data analysis pipeline as a microservice that collectively represent the overall application.

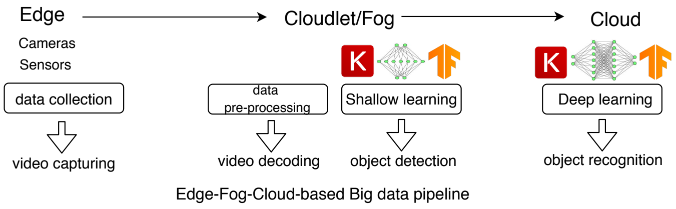

Fig. 2: Data analysis pipeline.

V. Multi-criteria bin packing optimisation

The multi-criteria, bin packing optimisation method was designed to provide a Pareto-based trade-off analysis for an efficient deployment of microservices along Edge-Fog-Cloud environment based on NFRs. As a result, the method returns a reduced number of options, Pareto non-dominant solutions which present the optimal infrastructure deployment options for running services. The implemented method fits into software engineering life-cycle of cloud-native applications, particularly as part of the service provisioning stage, where the deployment configuration is proposed, based on steps illustrated in Figure 3. In the following subsections all phases that are part of the method are introduced in details.

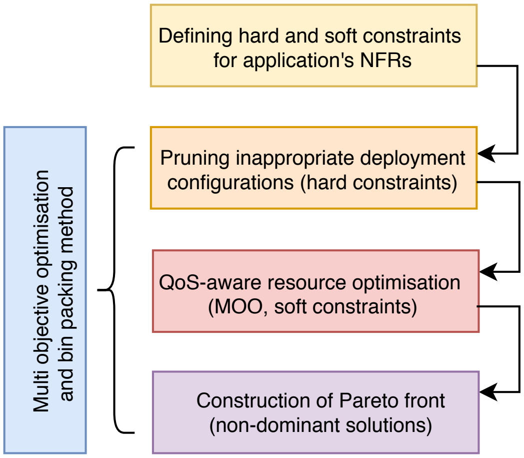
Defining hard and soft constraints for application's NFRs
Pruning inappropriate deployment configurations (hard constraints)
QoS-aware resource optimisation (MOO, soft constraints)
Construction of Pareto front (non-dominant solutions)
Multi objective optimisation and bin packing method

Fig. 3: The concept diagram of the Multi-criteria bin packing optimisation approach for trade-off analysis.

A. Defining hard and soft constraints

During the creation of cloud-native applications, a developer decides which non-functional requirements (NFR) belong to hard and soft constraints. Hard constraints are mandatory due to their essential influence on service operation, therefore they must be satisfied during runtime. On the other hand, soft constrains are not mandatory and tend to improve QoS. NFRs can be understood as parameters with categorical, continuous and ordinal values [14] and as such are fit for statistical analysis. In the current study, we have considered the following metrics, presented in Table I.

TABLE I: QoS metrics and their description.

Metric	Description
Latency	Total round trip time from user to app.
CPU utilisation	Computational capacity needed to process requests
Memory utilisation	Size of memory needed to process requests
Operational cost	Monetary cost of instances in the cloud (USD/h)
Geo-location	Physical location of the user (IoT device)

B. Multi-criteria bin packing optimisation approach

1) Pruning inappropriate deployment configurations based on hard constraints: Using a bin packing algorithm, we first remove all available instances in fog nodes and cloud data centres that do not satisfy hard QoS constraints associated with the application. For example, if the running service needs 3 CPU cores and 7 GB of memory, our approach prunes all instances which have an insufficient number of allocated compute resources, preferring those with a greater number of compute resources. Additionally, if the running service has a utilisation of 85% or greater of computing resources on one instance over a sustained time period, then this can result in service or system (e.g. instance) failure. As a consequence, the method is re-executed, searching for instances with a greater number of computing resources.

2) QoS-aware resource optimisation based on soft constraints: As the hard constraints are satisfied, for the QoS-aware optimisation we have utilised multi-criteria optimisation for fine tuning the optimal choice of suitable instances based on soft constraints. However, soft constraints can be mutually conflicting, whereby altering one attribute can have a detrimental effect on the others. For example, increasing service availability means providing system redundancy and consequently leads to higher operational costs. On the contrary, reducing operational cost means limited options for renting computational resources which can cause execution overheads [15].

C. Designing Pareto front non-dominant solutions

Our approach uses the concepts of domination to consider multiple conflicting NFRs. For $a \in A$ *dominates over* $a' \in A$ if a' is greater than a in relation to all objective functions, while a' has worse value for at least one solution. A solution $a' \in A$ is *non-dominated*, if there are no other solutions $a \in A$ that dominate over a'. A set of solutions $P' \in P$ is called *Pareto optimal* set if there are no other solutions in P that dominate any solution in P'. The set of all Pareto optimal solutions is known as *Pareto front*. The Pareto front is an efficient tool for supporting decision making. It narrows down the search space, and provides insights for efficient exploitation of the space of non-dominated solutions [15], [16].

D. Applying the method to the data pipeline

The deployment of workflow pipeline functionalities across Edge-Fog-Cloud environment is not a trivial procedure. We consider both coupled and interdependent (micro)services that exchange data. The deployment plan for the entire application could be generated in succession for one microservice after another, similarly, as the functionalities are executed in the pipeline. We propose 2 main phases as part of our method.

Phase 1: For the first workflow pipeline functionality (e.g. data-processing) as a microservice our approach proposes deployment plan. Based on it's initial requirements on compute resources and developer's desire on soft constraints (V-B) the approach returns a Pareto front, presenting optimal infrastructure options at fog nodes or cloud data centres. The developer makes a decision by selecting one Pareto point.

Phase 2: After proposing a plan for the deployment of the first workflow pipeline functionality, we consider the second and every next functionality (e.g. object detection and recognition) as a separated functionality yet dependent and linked to the first or previous functionality. For the generation of additional workflow pipeline functionality, our approach considers the selected Pareto point from *phase 1* and generates

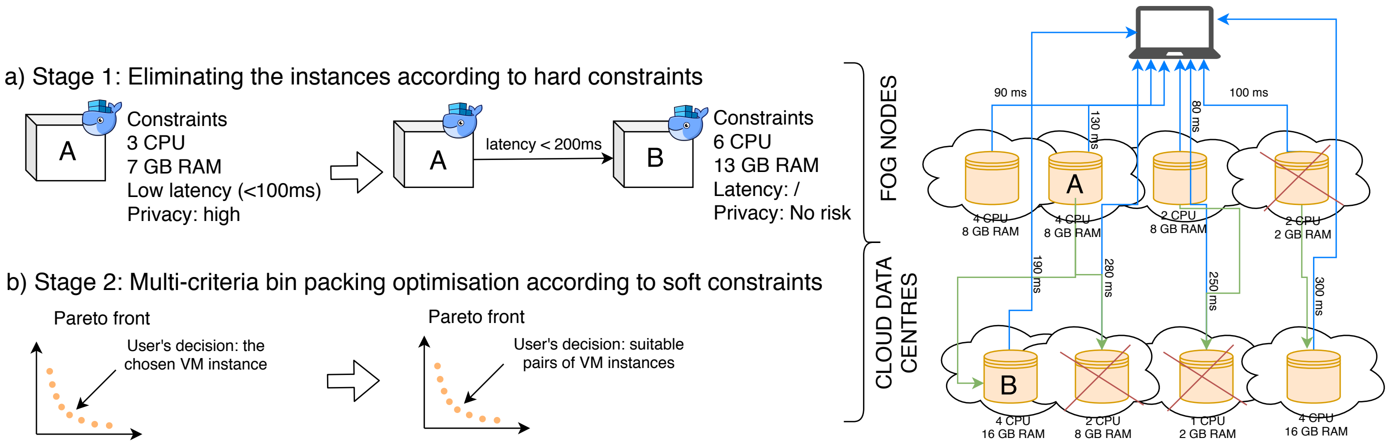

Fig. 4: Applying multi-criteria bin packing optimisation method to the Big data pipeline in two stages for deployment along Fog nodes and Cloud data centres

Pareto front. Each solution defines infrastructure pairs for the deployment of the second functionality whereas the approach considers latency from every fog node and cloud data centre in relation to the chosen node (selected Pareto point) in the *phase 1*. The advantage of this procedure is to narrow down the search space if we consider relationship and dependency among infrastructures in distributed fog nodes and clod data centres based on latency and geo-location. The two-phase process is depicted in Figure 4.

VI. Experimental Setup

We have simulated the traffic management scenario by running benchmark tests on 15 different Amazon's Elastic Compute Cloud (EC2) regions worldwide[1] including edge locations and data centres offered by EC2 [17] and considered two specific modules: (i) Client module and (ii) Connector Module (Cloud data centre).

The Client module collects the data from sensors (PPL), such as information on traffic and lane state and stores the data into database microservice. On each EC2 region, we have configured a test t2.micro instance and deployed the database microservice. The query requests were send from every test t2.micro instance to all test instances on various regions whereby we have measured round trip time (RTT) or latency of the query response. Additionally, to extend the experiments and consider a variety of other constraints, such as infrastructure-based metrics (CPU and memory utilisation) and operational cost of an instance we have configured several EC2 instances with different amount of allocated compute resources and on-demand operational cost. We have sent query requests from every test instance to all other test instances in all regions and measured the round trip time (RTT) of the query request and response. The collection of the monitoring data took more than a month whereby queries were sent every hour. For simplification of our experiments, we assumed that the RTT from specific instance at some region to the instances at the same region should be the same since it does not correlate with other compute properties of the instances, such as the amount of CPU and memory utilisation.

We then created a simple data-preprocessing microservice with initial constraints on CPU and memory utilisation and user constraints on latency and operational cost. Our main goal was to find the optimal infrastructure – a Client or Connector Module for the deployment of data pre-processing microservice based on hard (CPU memory utilisation) and soft constraints (latency, budget).

VII. Results and Evaluation

A. Results presentation

For our particular problem of Pareto front construction we have utilised the JMetal multi-objective optimization framework [18]. In Figure 5, the Pareto front non-dominant solutions are illustrated. As it can be seen from the chart, the Pareto front is constructed on the basis of two conflicting objectives, namely, operational cost on the x axis and RTT (or latency) of the query requests and response. The inappropriate instances based on hard constraints have already been removed in the first stage of our approach. The result is a construction of the Pareto front, which offers the optimal solutions in relation to both conflicting objectives. Moreover, the method reduces the number of all available VM instances to only the ones that provide optimal balance between operational cost and latency [15]. The tool has been created in a generic way, allowing to be extended to more than two objective functions.

B. Evaluation

For the sake of the evaluation, we have created a random deployment plan system by choosing random solutions and compared its performance against our approach. We have generated deployment plan 50 times whereas we still embedded basic logic into random deployment plan system in order to avoid making it too dummy by considering more amount of allocated compute resources as the service needs to avoid under provisioning. The summary of the evaluation is illustrated in Figure 6. The charts present Pareto-based and randomly chosen solutions (instances) according to the

[1] The following data centres have been utilised: N. Virginia, Ohio, N. California, Oregon, Mumbai, Seoul, Singapore, Sydney, Tokyo, Montreal, Frankfurt, Ireland, London, Paris, São Paulo.

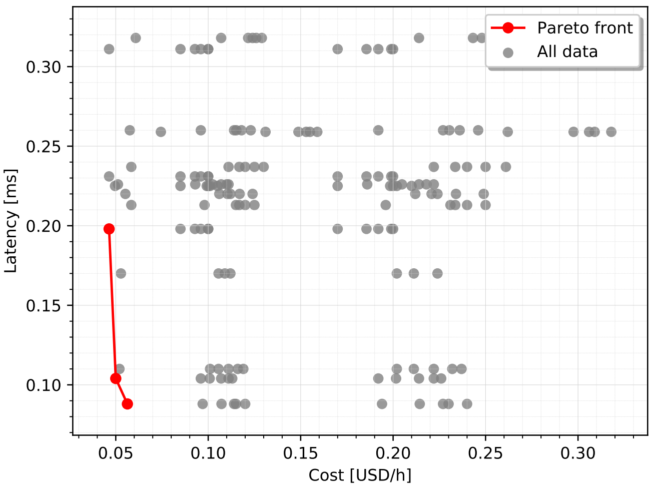

Fig. 5: Pareto front non-dominant solutions.

CPU and memory constraint. For the two constraints (CPU, memory), we assumed that the service operates well with 1 CPU core and 4 GB of memory. Nevertheless, our newly developed approach beats the random deployment plan system by (1) allocating as minimum amount of computing resources needed under the consideration of hard constraints and by returning the optimal solutions according to two conflicting objectives. On the contrary, the system for random instance choice considers instances with sufficient amount of compute resources but misses optimal allocation of compute resources and chooses the instances with much more resources which means over provisioning with higher operational costs and unsustainable consumption of compute resources.

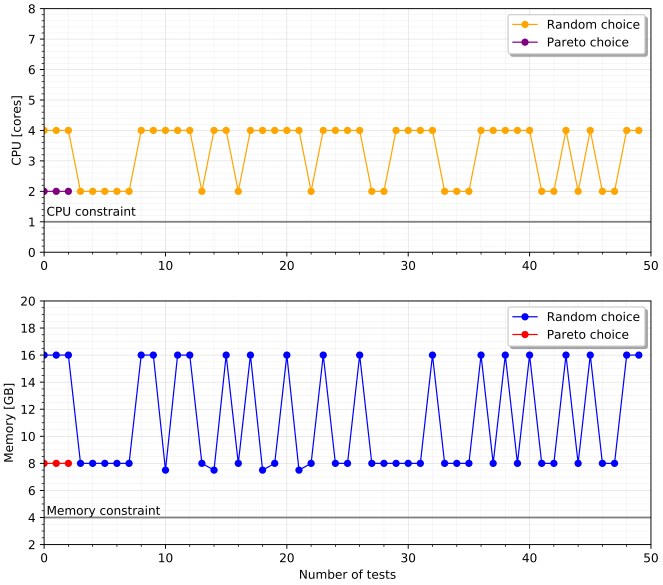

Fig. 6: Line charts depict pareto and randomly chosen solutions (VM instances). The horizontal line illustrates constraint.

VIII. Implication to Science Gateways

The presented approach can be used for the efficient distributed deployment of big data pipeline functionalities along edge, fog and cloud tiers with consideration of non-functional requirements. Supporting job placement and submission to edge resources enables extension to existing Science Gateways, which often make use of cloud-based resources for execution. Supporting close-to-the-user enactment of jobs and data placement enables approximate analysis and what-if investigations to be carried out at lower latency, compared to execution on a remote data centre.

A Science Gateway can also make use of the proposed approach to deploy services and support data migration in an automated manner. As a gateway can provide unified access to a number of different types of cyberinfrastructure and application libraries, the presented approach enables edge and fog devices to also be integrated in a seamless manner within such an environment. The optimisation approach outlined in this work enables dynamic: (i) deployment of services at a cloud data centre and at edge resources concurrently, based on requirements that have been identified by a user via a Web-based interface; (ii) auto-scaling of services across edge and cloud resources based on constraints that have been identified by a user.

The optimisation approach identified in this work can be used to schedule and execute tasks to a particular timeline and to specific resources. In the centrepiece is a Gateway where a user submits tasks e.g. (data pipeline functionalities) and the system (with the present approach integrated) automatically identifies the number of instances and resources needed to execute the tasks. The user would be able to manage tasks and resources through a user-friendly interactive web-based portal for automated job submission, such as [19] – which primarily makes use of a Kubernetes environment.

IX. Conclusion

With the emergence of Internet of Things (IoT) that periodically generate a massive amount of raw data, the need for processing and analysing of this data has become a challenge. Additionally, IoT applications that consist of workflow pipeline can benefit promptly from data processing near network edge due to low latency and preserved data privacy. On the other hand, large-scale cloud data centres offer high compute resources, scalability and availability. Due to the pipeline-based nature of IoT applications they are suitable for the distribution across edge, fog/cloudlet, cloud tiers.

We treated workflow pipeline functionalities as linked microservices that can be deployed one after another in the pipeline manner and jointly compose a fully working application. As an example, we provided the description of traffic management application scenario that illustrates, how workflow pipeline can be modelled and deployed to edge-fog-cloud infrastructure.

As the core of our work, we presented a Pareto-based Multi-criteria bin packing optimisation approach suitable for assuring a sufficient amount of mandatory compute resources and budget- and performance-efficient trade-off analysis of conflicting Non-functional Requirements. The experiments were run on Amazon's Elastic Compute Cloud (EC2) platform, particularly on 15 edge and data centre regions worldwide. The result of our approach are Pareto front non-dominant solutions that present an optimal infrastructure (e.g. instance) within fog nodes or cloud data centres for optimal deployment of the data-processing microservice with set initial constraints. Pareto front provides insights for easier exploitation of the search space and allows certain properties of particular interest to be easily explored. The method was evaluated in terms of operational budget and network latency benefits against the algorithm for random infrastructure choice. The results show that the present approach indeed returns optimal solutions and consequently the efficient deployment strategy plan. We estimate that the use of the present approach will encourage sustainable consumption of on-demand compute resources and provide enhanced Quality of Service of the overall application. Additionally, the present approach can be integrated into software engineering workbenches for the creation and deployment of cloud-native applications [20].

Our future research directions will be focused on orchestration, such as how to map the proposed deployment plan as part of the overall application workflow to the OASIS Topology and Orchestration Specification for Cloud (TOSCA) and dynamically update TOSCA. We will explore how data learning models can be executed in parallel and consequently horizontally and vertically distributed across edge, fog and cloud infrastructure.

Acknowledgment

The project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under grant agreement No. 815141 (DECENTER: Decentralised technologies for orchestrated Cloud-to-Edge intelligence).

References

[1] G. Burrows, "Harnessing the Internet of Things for business benefit," *The Nation*, 2015. [Online]. Available: https://images.forbes.com/forbesinsights/pitney_bowes_iot/HarnessingTheInternetofThings.pdf

[2] D. Evans, "The Internet of Things - How the Next Evolution of the Internet is Changing Everything," *CISCO white paper*, no. April, pp. 1–11, 2011. [Online]. Available: http://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:The+Internet+of+Things+-+How+the+Next+Evolution+of+the+Internet+is+Changing+Everything{#}0

[3] P. Mell and T. Grance, "The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology," *National Institute of Standards and Technology, Information Technology Laboratory*, vol. 145, p. 7, 2011. [Online]. Available: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

[4] Y. Simmhan, S. Aman, A. Kumbhare, R. Liu, S. Stevens, Q. Zhou, and V. Prasanna, "Cloud-based software platform for big data analytics in smart grids," *Computing in Science Engineering*, vol. 15, no. 4, pp. 38–47, July 2013.

[5] P. A. Legg, D. H. S. Chung, M. L. Parry, R. Bown, M. W. Jones, I. W. Griffiths, and M. Chen, "Transformation of an uncertain video search pipeline to a sketch-based visual analytics loop," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2109–2118, Dec 2013.

[6] Q. Pu, G. Ananthanarayanan, P. Bodik, S. Kandula, A. Akella, P. Bahl, and I. Stoica, "Low latency geo-distributed data analytics," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, ser. SIGCOMM '15. New York, NY, USA: ACM, 2015, pp. 421–434. [Online]. Available: http://doi.acm.org/10.1145/2785956.2787505

[7] M. Villari, M. Fazio, S. Dustdar, O. Rana, and R. Ranjan, "Osmotic computing: A new paradigm for edge/cloud integration," *IEEE Cloud Computing*, vol. 3, no. 6, pp. 76–83, Nov 2016.

[8] F. Bonomi, R. A. Milito, P. Natarajan, and J. Zhu, "Fog computing: A platform for internet of things and analytics," in *Big Data and Internet of Things*, 2014.

[9] Y. Wu, C. Wu, B. Li, X. Qiu, and F. C. M. Lau, "Cloudmedia: When cloud on demand meets video on demand," in *2011 31st International Conference on Distributed Computing Systems*, June 2011, pp. 268–277.

[10] A. R. Zamani, M. Zou, J. Diaz-Montes, I. Petri, O. Rana, A. Anjum, and M. Parashar, "Deadline constrained video analysis via in-transit computational environments," *IEEE Transactions on Services Computing*, vol. PP, pp. 1–1, 01 2017.

[11] G. Ananthanarayanan, P. Bahl, P. Bodík, K. Chintalapudi, M. Philipose, L. Ravindranath, and S. Sinha, "Real-time video analytics: The killer app for edge computing," *Computer*, vol. 50, no. 10, pp. 58–67, 2017.

[12] L. Knight, P. Stefanic, M. Cigale, A. C. Jones, and I. J. Taylor, "Towards a methodology for creating time-critical, cloud-based CUDA applications," in *Proceedings of IT4RIs 18: Interoperable infrastructures for interdisciplinary big data sciences- Time critical applications and infrastructure optimization*, Amsterdam, 2018. [Online]. Available: https://doi.org/10.5281/zenodo.1162877

[13] O. Rana, M. Shaikh, M. Ali, A. Anjum, and L. Bittencourt, "Vertical workflows: Service orchestration across cloud amp; edge resources," in *2018 IEEE 6th International Conference on Future Internet of Things and Cloud (FiCloud)*, Aug 2018, pp. 355–362.

[14] M. Kassab, M. Daneva, and O. Ormandjieva, "Scope management of non-functional requirements," in *Proceedings of the 33rd EUROMICRO Conference on Software Engineering and Advanced Applications*, ser. EUROMICRO '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 409–417. [Online]. Available: http://dx.doi.org/10.1109/EUROMICRO.2007.53

[15] P. Štefanič, D. Kimovski, G. Suciu, and V. Stankovski, "Non-functional requirements optimisation for multi-tier cloud applications: An early warning system case study," in *2017 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computed, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, Aug 2017, pp. 1–8.

[16] D. Kimovski, N. Saurabh, V. Stankovski, and R. Prodan, "Multi-objective middleware for distributed vmi repositories in federated cloud environment," *Scalable Computing: Practice and Experience*, vol. 17, no. 4, pp. 299–312, 2016.

[17] A. W. Services, "Amazon Elastic Container Service Developer Guide," 2014. [Online]. Available: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/ecs-dg.pdf

[18] N. A. J. Durillo, J. J., "jmetal: A java framework for multi-objective optimization," *Advances in Engineering Software*, vol. 42, no. 10, pp. 760–771, 2011.

[19] GitHub, "Argoproj," online; accessed 6 May 2019. [Online]. Available: https://argoproj.github.io/argo/

[20] P. Stefanic, M. Cigale, A. C. Jones, L. Knight, I. Taylor, C. Istrate, G. Suciu, A. Ulisses, V. Stankovski, S. Taherizadeh, G. Flores Salado, S. Koulouzis, P. Martin, and Z. Zhao, "SWITCH workbench: A novel approach for the development and deployment of time-critical microservice-based cloud-native applications," *Accepted to Future Generation Computer Systems*.