Challenges and Research Directions in Big Data-driven Cloud Adaptivity

Andreas Tsagkaropoulos[1], Nikos Papageorgiou[1], Dimitris Apostolou[1,2], Yiannis Verginadis[1] and Gregoris Mentzas[1]
[1]*Information Management Unit (IMU), Institute of Communications and Computer Systems, National Technical University of Athens (NTUA), Athens, Greece*
[2]*Department of Informatics, University of Piraeus, Piraeus, Greece*

Keywords: Cloud Adaptivity, Fog Computing.

Abstract: Mainstream cloud technologies are challenged by real-time, big data processing requirements or emerging applications. This paper surveys recent research efforts on advancing cloud computing virtual infrastructures and real-time big data technologies in order to provide dynamically scalable and distributed architectures over federated clouds. We examine new methods for developing cloud systems operating in a real-time, big data environment that can sense the context of the application environment and can adapt the infrastructure accordingly. We describe research topics linked to the challenge of adaptivity such as situation awareness, context detection, service-level objectives, and the capability to predict extraordinary situations requiring remedying action. We also describe research directions for realising adaptivity in cloud computing and we present a conceptual framework that represents research directions and shows inter-relations.

1 INTRODUCTION

Cloud computing and, in particular, the Infrastructure-as-a-Service (IaaS) model has become the natural habitat for big data driven applications, thanks to its unlimited scaling abilities. Big data applications challenge cloud computing infrastructures which have not been designed to cope with the velocity and sheer quantity of generated data. The challenge is even greater when cloud infrastructures are called to deal with the immense amount of data that the billions of distributed IoT sensors can genera-te. Thus, the cloud computing paradigm is being expanded to reach the extreme edge of the network, effectively developing so-called fog infrastructures.

Mainstream technologies, such as Hadoop, are challenged by real-time processing requirements, due to their static nature (Burns 2013). While more recent and dynamic technologies, such as Storm (Apache Storm 2018), have proven to be efficient computational platforms for real-time data stream analytics, they fail to exploit the relatively new decentralized architecture known as federated clouds and fog infrastructures. A federated cloud is a hybrid cloud that merges the resources available in multiple clouds. Federated clouds consist of resource which can be private clouds (e.g. in-premises OpenStack installation), public IaaSes (e.g. Amazon Web Services, Microsoft Azure) or decentralized clouds (e.g. fog and edge clouds). Cloud resource configuration can be performed using open standards and APIs such as OCCI (OCCI, 2018), CIMI (CIMI, 2013), TOSCA (OASIS, 2017) or proprietary but widely used APIs like Amazon Web Services (Amazon, 2018).

Recent research efforts have focused on advancing cloud computing virtual infrastructures and real-time big data technologies in order to provide dynamically scalable and distributed architectures on federated clouds. Various works have proposed new methods for developing cloud systems operating in a real-time, big data environment that can sense the context of the application environment and can adapt the infrastructure accordingly.

Adaptivity in big data-driven cloud infrastructures manifests primarily in two layers (Figure 1): The real-time application adaptivity l ayer refers to changes that can be performed in real time by application-specific code on end devices.

Tsagkaropoulos, A., Papageorgiou, N., Apostolou, D., Verginadis, Y. and Mentzas, G.
Challenges and Research Directions in Big Data-driven Cloud Adaptivity.
DOI: 10.5220/0006761901900200
In *Proceedings of the 8th International Conference on Cloud Computing and Services Science (CLOSER 2018)*, pages 190-200
ISBN: 978-989-758-295-0

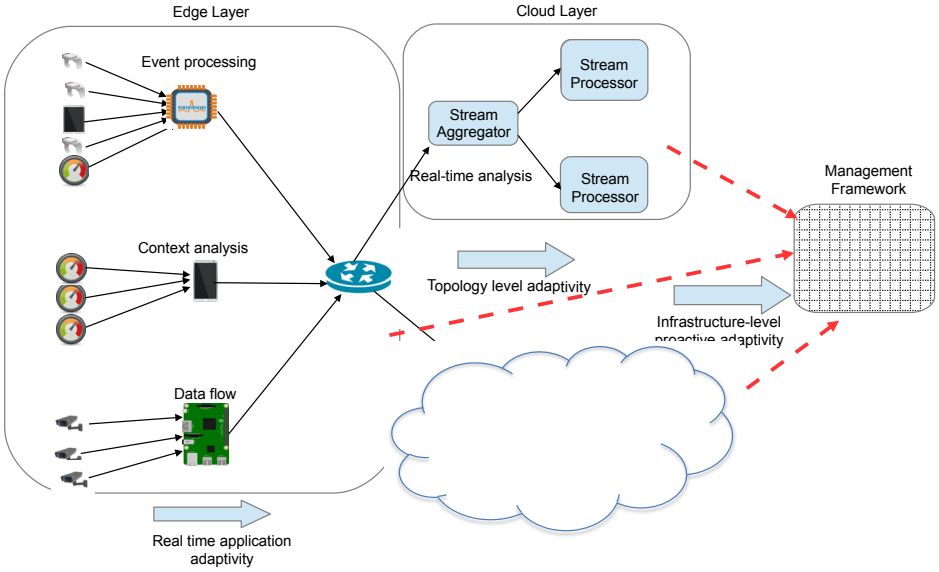

Figure 1: Adaptivity in big data-driven cloud infrastructures.

The topology adaptivity layer refers to changes that can be performed to the processing topology and the virtual resources available.

The challenge of adaptivity is linked to a number of research topics that we survey in this paper: situation awareness, context detection and the capability to predict situations that pose the need for adaptation. Our research objective is to investigate the related state of the art, and facilitate the work of researchers and practitioners through the presentation of a conceptual framework. The proposed framework aims to identify the tools and methods needed for optimizing (i) the resource usage for various applications based on specific policies and (ii) the quality of service demands of specific application ecosystems.

The remaining of the paper is structured as follows: Section 2 provides an overview of the state of the art in cloud adaptivity. Section 3 identifies challenges for cloud computing technologies and related research directions. Section 4 presents a meta-management framework for cloud adaptation for guiding relevant research & development needed to realise big-data driven cloud adaptivity. Section 5 concludes the paper.

2 STATE OF THE ART

Cloud adaptation refers to the process of dynamic selecting and configuring the cloud resources (such as CPU, memory, storage or networking) that are necessary to provide a service with the desired quality attributes. (Jennings and Stadler, 2015) point out that in cloud-computing there exist different types of actors which have different objectives pertaining cloud adaptation needs. Cloud providers provide services (e.g. cloud infrastructure) to cloud users (e.g. DevOps). Cloud users provide services to end users (e.g. a web-based application). In IaaS or PaaS context, cloud providers agree on certain SLAs and Service Level Objectives (SLOs) with cloud users. There are many categories of SLAs (Kyriazis, 2013). According to the authors, more relevant to resource management are the SLOs that are quantifiable (like those that are related to performance and availability). SLOs sometimes, depending on the definition of the SLA, may be satisfied up to a certain possible degree, when other constraints must be satisfied simultaneously.

Cloud providers may pursue additional objectives that are important for their business such as energy use minimization and fault tolerance, in some cases in a prioritized manner depending on the context (e.g. prioritize low energy use when the total workload is not high). Cloud users have SLAs with their customers. In order to satisfy them they agree SLAs with cloud providers but they may have additional objectives (such as minimizing the risk of violating SLAs with end users by using multiple cloud providers).

(Jennings and Stadler, 2015) in a recent survey of over 250 publications about resource management relevant to cloud based infrastructures recognize four types of adaptation: “Infrastructure Scaling”, “Virtual Machine Migration”, “Virtualization” and

"Equipment Power State Adjustment". Challenges that the authors identify include: (i) the way to achieve predictable performance when adaptations are performed, knowing that the cloud resources are shared between different applications; and (ii) the way to globally manage different types of resources (compute, storage, networking, etc.) in an orchestrated way in order to achieve "Global Manageability of Cloud Resources". (Ranjan et al.,2015) summarize the different types of attributes (cores, speed, capacity, etc.) that can be configured and orchestrated in cloud infrastructure resources (CPU, BLOB storage, network, etc.) according to its type (IaaS, SaaS, PaaS) and the operations that can be applied to each attribute (start, stop, restart, etc.). This work not only describes the cloud resources as the ontology in (Youseff et al.,2008) but it attempts to identify the way a cloud-based infrastructure can be programmed in order to achieve its goals.

2.1 Adaptation Implementation

Concerning the actual implementation of adaptation actions, proposed approaches broadly fall under two categories: *Proactive* adaptation policies - which anticipate unusual situations and take actions in order to ensure the correct functioning of the service and *reactive* adaptation policies, which use data from the current situation in order to trigger the necessary adaptation actions.

The review paper of (Lorido-Botran et al.,2014) identifies and compares five categories of approaches for decision making in auto-scaling: Threshold-based rules, Reinforcement Learning (RL), Queuing theory (QT), Control theory (CT) and Time series analysis (TS). (Gandhi et al., 2012) identify five similar auto-scaling approach categories: Prediction models, Control theoretic techniques, Queueing-based models, Black-box and Grey-box approaches. Black box models use machine learning or statistical methods for decision making in order to overcome the problem of modelling the cloud application using expert knowledge. Grey-box models are hybrid approaches that use models in combination with machine learning (Gandhi et al., 2012). (Qu et al., 2016) support in their review paper that, according to the bibliography, resource estimation in horizontal or vertical auto-scaling can be performed using Rules, Fuzzy-Inference, Application-Profiling, Analytical Modelling, Machine Learning or hybrid methods. Analytical modelling according to the authors includes Queuing Theory and Markov Chains. Machine learning includes Reinforcement Learning and Regression. Regression is applied in auto-scaling techniques that use Time-series Analysis or Control Theory.

Rule-based approaches are used in commercial auto-scaling systems such as (Amazon AWS Auto-Scaling service,2018) or (RightScale, 2018). An advantage of this approach is that a DevOps user can relatively easily create and understand them. In (Lorido-Botran et al.,2014) the authors maintain that while rule-based approaches are usually reactive, they can be combined with queueing theory, or time-series analysis, in order to respond proactively to situations.

Another approach for adaptation relies on Queuing theory, which discusses queues (waiting lines) using mathematics. The modelling using queues is relevant to cloud applications, because the requests to a service can be assumed to follow a queue. The result of the modelling is the necessary resources to process a workload with a specific size. However, as queuing models require a fixed architecture in order to be solved, they need to be solved every time that an adaptation occurs which is computationally expensive.

A different approach uses Control Theory to maintain an output variable (e.g. CPU usage) constant, while modifying the resources of the cloud infrastructure respectively (e.g. number of VM's/characteristics of a VM). According to the review paper by (Lorido-Botran et al.,2014), these approaches are fine-tuned either for vertical scaling or horizontal scaling, but not for both.

An alternative method to handle the problem or resource adaptation relies on machine learning, and more specifically on Reinforcement Learning (RL). Using this method, an agent interacts with the system and proposes the best adaptation action (from the pool of defined actions) based on its experience in similar situations. Then, the outcome of the adaptation is measured using a cost/reward function, and the agent improves its understanding. Unfortunately, models following the RL method suffer from bad performance during on-line training for a long time, require considerable processing power, and – being highly specialized – perform sub-optimally when the context is changed(Lorido-Botran et al.,2014) .

Table 1 summarises the characteristics of the aforementioned cloud infrastructure adaptation approaches.

Table 1: Summary of Cloud Infrastructure Adaptation Approaches.

Adaptation method	Policy Type (proactive vs reactive)	Real-time model adaptation	Model response after context change	Vertical and Horizontal Scaling Optimization	Model update difficulty	Computational complexity
Reinforcement Learning	Proactive	Yes	Bad	Yes	Hard	High
Queueing Theory	Proactive	No	Bad	None	Hard	High
Control Theory	Reactive	No	Good	Only one attribute optimized	Hard	Average
Time Series Analysis /Prediction Models	Proactive	Yes	Very Good	None	Average	Average
Rule-based approaches	Reactive (Can be combined with queuing theory methods or prediction methods to gain proactivity)	No	Good	Yes	Average	Low

2.2 Application-specific Adaptivity

This section presents recent approaches that deal with the challenge of adapting applications than run on cloud infrastructures (see also Table 2). (Cavalcante et al., 2013) developed an autonomous adaptation process for cloud-based applications by replacing a service by an alternative one that fulfils the application needs and describe the adaptation process within the Cloud Integrator, a service-oriented middleware platform for composing, executing, and managing services provided by different cloud platforms. (Inzinger et al. 2013), (Inzinger et al, 2014), proposed a provider-managed, model-based adaptation approach for cloud computing applications, allowing customers to specify application behaviour goals or rules, thus actively engaging the application refactoring process. There are also recent agent-based efforts that try to fuse adaptivity in cloud resources usage. For example, (Comi et al.,2015) present an approach based on agent cloning, i.e. a mechanism of agent reproduction allowing providers to substitute an "unsatisfactory" agent acting in a "cloud context" with a clone of an existing agent having suitable knowledge and good reputation in the multi-cloud context.

Another stream of work focuses on application migration approaches. (Gholami et al., 2016) published a detailed survey of cloud migration approaches. The authors revealed that little work exists that provides a means to design situation-specific approaches with respect to the characteristics of a migration project. (Inese et al., 2015) described methods used for enterprise application decomposition in cloud migration projects. Their work distinguishes between four decomposition phases: fact extraction, pre-processing, clustering/component classification and post-processing. Methods for fact extraction include static, dynamic and semantic code analysis as well as dynamic SQL analysis. Pre-processing includes similarity evaluation, trace compression, rules, classification, code cleansing and concept assignment. Clustering and component identification is typically done with clustering methods as well as rules. Finally, post-processing refers to evaluation methods using rules, refinement as well as optimisation and layer identification.

A methodological approach for cloud migration has been developed by (Jamshidi et al., 2016) based on (i) a catalogue of fine-grained service-based cloud architecture migration patterns that target multi-clouds, (ii) a Situational migration process framework to guide pattern selection and composition, and (iii) a variability model to structure system migration into a coherent framework. The proposed migration patterns are based on empirical evidence from several migration projects, best practice for cloud architectures and a systematic literature review of existing research. The methodology allows an organization to (i) select

appropriate migration patterns, (ii) compose them to define a migration plan, and (iii) extend them based on the identification of new patterns in new contexts.

A cloud migration strategy recommender method is proposed by (Bonab and Busherian), 2015). The main rationale behind this method is to provide easy and fast recovery from failed components or replacing the required functionality of the legacy software with the reliable cloud services. In this paper a semi-automated reverse engineering method based on the clustering algorithms is proposed to recommend the best migration-to-cloud strategy. The recommendation is based on four defined metrics: the extent of effort required for reengineering, maintenance costs, achieved availability and the number of cloud services that are used.

(Kwon and Tilevich, 2014) proposed an application refactoring approach based on a recommender tool that computes the coupling metrics for all the classes in a legacy application and then displays the classes that are least tightly coupled. Accessing the functionality represented by these classes from a remote cloud-based service should impose only a limited performance penalty on the refactored application. The approach leverages two recommendation mechanisms: profiling and clustering-based recommenders. The profiling-based recommender engages a static program analysis and runtime monitoring to collect program information. By combining the class coupling metrics collected through both static analysis and runtime monitoring, the recommendation algorithm then suggests a subset of an application that can be transformed to cloud-based services. The profiling-based recommender sorts application classes based on their execution duration and frequencies, so that the programmer can know what classes are computation-intensive and how frequently they are accessed. (ii) The clustering-based recommender clusters classes with similar functionality, thus identifying class clusters whose functionality can be naturally exposed as a cloud-based service. Because the clustering-based recommender groups classes based on their functionality, the programmer can avoid duplicating a functionality in the cloud by selecting candidates for cloud-based service from different clusters.

(Hilton et al., 2014) developed Cloudifyer, a touchdevelop IDE plugin which refactors touchdevelop scripts in place. First, Cloudifyer retrieves the source of the target app as an Abstract Syntax Tree (AST) stored in JSON format from the touchdevelop script bazaar. It then transforms the AST as needed. Once all the transformations are performed, Cloudifyer completes the refactoring by saving the new AST for the target app. (Vasconcelos et al.,2015) presented a novel approach to support organizations in automatically adapting existing software applications to the cloud. The approach is based on the loosely-coupled implementation of non-intrusive code transformations, called cloud detours, which enable the automatic replacement of local services used by an application with similar or functionally-related services available in the cloud.

Table 2: Summary of Application-level Adaptivity Approaches.

Authors	Approach	Adaptation type	Active multi-cloud support
Cavalcante et al.	Replacement of a service with a new instance fulfilling application needs	Component replacement	No
Inzinger et al.	Adaptation based on application behavior goals or rules	Satisfaction of goals /rules	No
Comi et al.	Substitution of unsatisfactory agents with suitable clones of better-performing existing agents	Component replacement	Yes
Inese et al.	Methods for application decomposition in cloud migration projects	Cloud migration	Yes
Jamshidi et al.	Methodological approach for cloud migration	Cloud migration	Yes
Bonab et al.	Easy and fast recovery from failed components/ replacement of legacy components with reliable cloud services	Cloud migration	Yes
Kwon and Tilevich	Application refactoring based on class-coupling metric	Processing offloading to the Cloud	No
Hilton et al.	Refactoring of touchdevelop scripts in place	Processing offloading to the Cloud	No

3 CHALLENGES AND RESEARCH DIRECTIONS

The majority of existing works in cloud adaptation focus on cloud resource selection and configuration based on the computing needs of applications but do not take into consideration trade-offs between cost, speed, efficiency and reliability of adaptations. Future research can investigate new ways to manage the relations between Big Data processing, cloud and edge resources adaptation needs and adaptation strategies. Novel data analytics approaches are needed for understanding from the past data different modes of the cloud infrastructure workloads and predicting which can be the next one based on the current context of big data applications. Factors that influence the workload can be studied and new behavioural models can be built from past data. These models should be truly data driven and built in an unsupervised manner (no a priori labelled data should be used).

Approaches for analysing past data should in a way 'explain' what the usual behaviour of the system is in a high-dimensional space, which can dynamically change as incoming data will change. Methods should ensure that the models will be continuously updated to the new situations, resolving the problem of the model drift. Moreover, models should be used in real-time in order to check what the current parameters of the system indicate regarding the workload, i.e. can the current situation be classified as "usual" (with the standard set of actions), or it is unusual and requires an additional processing.

Moreover, new methods are needed for extracting and modelling context features of big data applications that pertain cloud resource requirements. To this end, works may need to extend decision methods (Patiniotakis et al.,2013a), (Patiniotakis et al.,2013b) with the necessary enhancements and extensions in order to deal with the scalability issues involved in the Big Data processing application domain. This will enable adapting resources allocation and deploying parts of the data-intensive applications across the processing topology. For example, a new processing resource can be recommended based on a predefined but dynamically changeable pool of resource alternatives that should be configurable according to the context of the case. Moreover, the processing load of a specific resource can be offloaded to another one that maybe closer to the edge of the network for efficiency reasons.

To support adaptation triggering, instead of using static threshold-based alerts for an entire class of cloud resources as most monitoring systems typically use, there is a need to develop new triggering methods based on monitoring multiple data streams, understanding each individually as well as their relationship to each other, resulting in a highly sensitive system that can provide early recommendations for adaptations to optimise performance.

4 ADAPTATION META-MANAGEMENT FRAMEWORK

This section describes our proposal for a meta-management framework for big data-driven cloud adaptation. This refers to a conceptual layer on top of the existing cloud infrastructure management layers aiming to provide support for dynamic adaptation of the cloud infrastructure based on dynamically changing, big data application computing needs. The optimal selection of cloud resources is the primary objective of the meta-management framework, which should support decisions making for the number and kind of resources, the location of the resources and the level of virtualization. The so-called decision enablers are described next.

4.1 Adaptivity Decision

4.1.1 Scale Up - Scale Down

The first function of the proposed framework is to scale the cloud application service up and down, depending on the workload, using a suitable deployment platform (virtual machines or containers). This scaling can be implemented horizontally or vertically (Michael et al., 2007). Horizontal scaling is defined as the ability of a cloud service to spawn more (limit existing) VM instances running the same application. Vertical scaling is defined as the process of increasing (decreasing) the specifications of the VM or container hosting the application at runtime. If an application can use both types of scaling, then we say that it can use hybrid scaling.

(Dutta et al.,2012) claim that vertical scaling can offer lower cost adaptations, most appropriate when the range of the intensiveness of the application workload is relatively small. Besides, horizontal

scaling can allow for much higher throughput albeit possibly at higher cost. (Fang et al., 2012) describe vertical scaling as a methodology most appropriate for regular adaptations to workload, whereas horizontal scaling is more appropriate in cases of sharp workload changes.

However, not all applications support vertical scaling and horizontal scaling. For example, applications relying on the JVM should restart before being able to modify the heap allotted to a java program. Also, applications not using external load-balancing frameworks will not benefit from the additional instances spawned. The above mentioned special characteristics of each mode of adaptation indicate that research is required by application developers in order to formulate the correct criteria to trigger adaptations. A relevant survey, which examines many scaling techniques used by dedicated software (auto-scaling) has been compiled by (Qu et al., 2016)

4.1.2 Migrate & Re-position

The second function of the framework is live service migration. Live migration is the transfer of a service to another processing resource, without user-noticeable service interruption. Not only does it enable maintenance operations to be conducted without the interruption of the service, but it also permits the reconfiguration of the service under unexpected workloads, and can be used as a preparation step for efficient vertical scaling. Several published studies examine live migration as a means to consolidate resources or handle increasing workloads (Bryant et al., 2011), (Hermenier et al., 2009), (Nguyen et al, 2013). It must be noted however that live migration degrades the performance of all virtualized entities running on both the physical machine offloading the VM/container, and the physical machine onloading the VM/container (Koto et al, 2012).

4.1.3 Application-specific Adaptation Actions

A third function of the framework is related to supporting the class of adaptation actions that consider the internal structure of the application.The framework should expose to the DevOp the capability to dynamically modify hardware resources with application-level logic. For example, in a grid computing cluster, the DevOp can specify that when a “multicore processing” job arrives, all processing nodes should request at least 8 cpu cores. This flexibility provides one more means for the DevOp to ensure proper quality of service.

4.2 Enabling Capabilities

To support technically decision making with respect to cloud adaptivity, the framework puts forwards certain enabling capabilities. In a real-world setting, each capability described below reflects a software component that can range in implementation from a simple process in a virtual machine to a distributed multi-cloud processing architecture. The enabling capabilities are depicted in Figure 2.

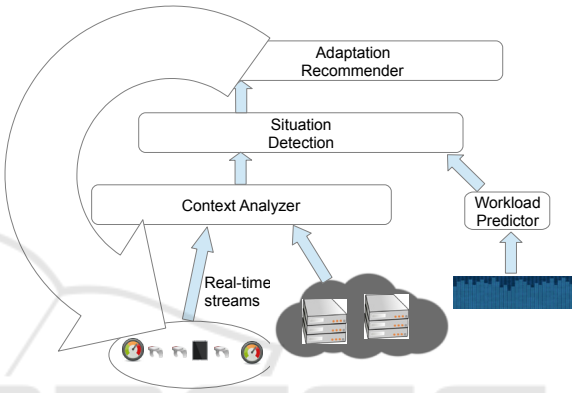

Figure 2: Adaptation meta-management framework.

4.2.1 Workload Prediction

The meta-management framework should have the capability, given an appropriate model and method, recent monitoring information and workload evolution over time, to predict the workload that may be experienced by the cloud infrastructure in the near future. In addition, it should be able to predict possible failures because of the overuse of certain processing nodes, thus enhancing the reliability of the processing topology and minimizing any chances for down time incidents.

While it is possible to predict the workloads of long-running tasks based on the seasonality in their historical workloads, it is difficult to do so for tasks which do not have such recurring workload patterns. Novel workload prediction approaches should be developed that take the statistical properties of a pool of tasks to help predict the workload patterns of new tasks. Efficient machine learning methods can process high-dimensional data and can be applied directly to memory, disk bandwidth, and network bandwidth demand predictions. Both machine learning and statistical methods can leverage readily existing tools, such as (TensorFlow, 2018), (Keras, 2018) or (R, 2018).

Workload prediction methods should be capable of

- Receiving, pre-processing and storing the data relevant for workload prediction (for example: CPU usage, RAM usage, network and disk i/o, etc.).
- Understanding and modelling the types of the workloads based on the provided data (infrastructure, applications).
- Predicting the workload of the underlying cloud infrastructure.
- Enabling refinement process (self-adaptation).

4.2.2 Situation Detection

Situation Detection capabilities should detect situations that might lead to resources adaptation or data-intensive application reconfiguration or redeployment. Situation Detection should receive as inputs the context of edge devices & cloud resources, the current workload (e.g. current throughput, volume) as well as workload predictions (e.g. predicted throughput, volume) and should generate as outputs the detected situation and its associated context conditions. Novel situation detection methods are needed that go beyond the state of the art by enabling smart situations detection based on event subscriptions, workload predictions capable of coping with the dynamicity of Big Data. Scalable, high-performance complex event processing engines (e.g (Siddhi, 2018)) can be used to implement the functionalities described above.

Situation Detection methods should be capable of:

- • Detecting situations during the usage of the Cloud Application that might indicate the necessity for adaptation under certain context conditions. For example, if a server node is indicated to have high CPU usage levels, then a situation requiring immediate action is reflected (because the service might start dropping requests). In the case described above, the detected situation is that a server machine is under severe load.
- • Triggering adaptation of the processing topology.
- • Guiding the deployment of application fragments.

4.2.3 Context Analysis

Efficient analysis of the status of cloud resources including edge devices should correlate between cloud and edge related attributes, types of jobs running and “observed” QoS variations which constitute the context about the current status of cloud and edge devices. For example, context may refer to the task execution time on an edge device or an estimate of the remaining battery lifetime for edge device where it is applicable. The main element necessary for context analysis is monitoring data, such as real-time QoS variations, cloud resource metrics and edge device state parameters – for example battery, location and network . This data will be transmitted by a special agent, present on all cloud resources and edge devices. Finally this data should be processed (e.g using machine learning and /or other statistical techniques) in order to identify the context of each resource/device and generate meaningful correlations between the monitored variables. Tools such as (Apache Kafka, 2018) can be used to manipulate high-volume streams of monitoring data.

Context Analysis should:

- Collect monitoring data from cloud resources and edge devices and infer context.
- Relate current context to the processing capacity of a device, e.g. expected execution time, battery level.

4.2.4 Adaptation Recommenders

We distinguish between two broad types of recommendations the framework should deliver: Recommendations for adaptations of the applications running on the cloud infrastructure and recommendations for cloud resource adaptations.

The first type of recommenders should produce, for example, the appropriate partitioning of cloud applications into smaller parts that can be efficiently deployed over cloud / edge resources. They should also associate applications and application parts with placement constraints and optimization preferences. For example, they could advise that application part 1 must run on a VM with RAM > 4 GB, application part 2 may run on any edge device, or that all parts should be placed under the same availability zone. Input may be the available VM flavours & edge devices as well as the qualitative, quantitative preferences of the DevOp or application developer in order to formulate the optimization function. Outputs may be a recommended partitioning of the application along with a recommended deployment serialized in a TOSCA-based specification (without specific VM and edge instances). The recommendations should take into consideration

constraints such as security constraints or other quantitative or qualitative constraints, e.g. cost, response time, data sanitization support etc.

The recommendations will be forwarded to the second type of recommenders, which will then use them to reactively (or proactively in case the situation detected involves workload predictions) find an optimal cloud resource reconfiguration. The proposed reconfiguration will include the changes on the used cloud resources, the used edge devices, and the placement of applications or application parts.

Inputs may be the current processing topology and placement, the detected situations along with the respective context of the used and the available edge devices. Output can be the *adaptation recommendation* to reconfigure the processing topology, e.g., to introduce new processing nodes, replicate nodes for failover purposes, remove redundant or underused processing nodes. Recommendations will entail how to alter the processing topology on cloud resources, e.g., to reconfigure additional VMs that host existing processing nodes, spawn VMs to new physical machines to deal with failover, start new containers or deploy on additional hosts in a cluster. The recommendation may be the product of any suitable decision technique. In the example provided before, a possible adaptation recommendation could be the instantiation of a new server instance, and a load-balancing service to alleviate the high CPU load.

Finally, recommendations will include shifting processing effort to/from resources at the extreme edge of the network, if necessary.

5 CONCLUSIONS

In this paper, we investigated challenges, research efforts and research directions related to the adaptation of next-generation cloud infrastructures to support the advanced processing needs of big data applications. We introduced the related challenges: situation awareness, context detection, adaptivity and the capability to predict situations adaptation. To the best of our knowledge, there is currently no approach that goes beyond cloud resource selection and allocation towards recommending cloud and edge resource adaptations based on the dynamically changing processing needs, taking under consideration trade-offs between cost, speed, efficiency and reliability of adaptations. Our work provides directions to researchers for investigating new ways to manage the relations between Big Data processing, cloud and edge resources adaptation needs and adaptation strategies.

The proposed framework aims to provide a blueprint for extending the state of the art through a novel data analytics approaches for understanding from the past data different modes of the workloads and predicting which can be the next one based on the current situation. Moreover, the framework emphasises the need for situation-awareness capabilities, which can be supported using event processing technologies.

With respect to adaptation recommendations, we plan to research and develop two recommenders that will issue recommendations with respect to adapting resources allocation in real-time and deploying parts of the data-intensive applications across the processing topology. For example, a new processing node can be recommended based on a predefined but dynamically changeable pool of alternatives that should be configurable according to the context of the case. Moreover, the processing load of a specific node can be recommended to be offloaded to another one that maybe closer to the edge of the network for efficiency reasons. For developing these recommenders, we propose methods for run-time evaluation of adaptation actions based on situations detected from the big-data processing and the current usage context. To support predictive behaviour, instead of using the same threshold-based alerts for an entire class of cloud resources as most monitoring systems use, we propose the development of systems that will be monitoring multiple data streams, understanding each individually as well as their relationship to each other, resulting in a highly sensitive system that can provide early recommendations for adaptations to optimise performance.

ACKNOWLEDGEMENTS

This work is partly funded by the European Commission project H2020 PrestoCloud - Proactive Cloud Resources Management at the Edge for Efficient Real-Time Big Data Processing (732339).

REFERENCES

Amazon AWS, 2018. AWS Cloud computing services. Available online: https://aws.amazon.com/.

Amazon AWS Autoscaling Service, 2018. Amazon Web Services (AWS), Auto scaling. Available online: https://aws.amazon.com/autoscaling/.

Apache Kafka, 2018. Apache Kafka, A distributed streaming platform. Available online: https://kafka. apache.org/

Apache Storm, 2018. Available online: http://storm.apache.org/.

Bonab, B. A., and Bushehrian, O., 2015. A semi-automated reverse engineering method to recommend the best migration-to-cloud strategy. In *2015 International Symposium on Computer Science and Software Engineering (CSSE), Aug. 2015, pp. 1–7.*

Bryant, R., Tumanov, A., Irzak, O., Scannell, A., Joshi, K., Hiltunen, M., Lagar-Cavilla, A., and de Lara, E., 2011. Kaleidoscope: Cloud micro-elasticity via vm state coloring. In *Proceedings of the Sixth Conference on Computer Systems, ser. EuroSys '11, ACM, 2011, pp. 273–286*.

Burns, E., 2013. Hadoop still too slow for real-time analysisapplications?. Available online: http://search businessanalytics.techtarget.com/feature/Hadoop-still-too-slow-for-real-time-analysis-applications.

Cavalcante, E., Batista, T., Lopes, F., Almeida, A., de Moura, A. L., Rodriguez, N., Alves, G., Delicato, F., and Pires, P., 2013.Autonomous adaptation of cloud applications. In *Distributed Applications and Interoperable Systems: 13th IFIP WG 6.1 International Conference, DAIS 2013. Springer, 2013, pp. 175–180.*

CIMI, ISO/IEC JTC 1, 2013.Cloud infrastructure management interface (cimi) model and restful http-based protocol, ISO/IEC19831. In *ISO Standards Catalogue.*

Comi, A., Fotia, L., Messina, F., Pappalardo, G., Rosaci, D., and Sarné,G.M. L.,2015. An evolutionary approach for cloud learning agents in multi-cloud distributed contexts. In *2015 IEEE 24th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, Jun. 2015, pp. 99–104*.

Dutta, S., Gera, S., Verma., A., and Viswanathan, B.,2012. SmartScale: Automatic application scaling in enterprise clouds. In *2012 IEEE Fifth International Conference on Cloud Computing, IEEE, pp. 221–228* (2012). doi:10.1109/CLOUD.2012.12.

Fang, W., Lu, Z., Wu, J., Cao, Z., 2012. RPPS: a novel resource prediction and provisioning scheme in cloud data center. In *2012 IEEE Ninth International Conference on Services Computing, IEEE, pp. 609–616 (2012)*. doi:10.1109/SCC.2012.47.

Gandhi, A., Dube, P., Karve, A., Kochut, A. and Zhang, L., 2014. Adaptive, model-driven autoscaling for cloud applications. In *11th International Conference on Autonomic Computing (ICAC14), USENIX, 2014, pp.57–64*.

Gholami, M. F., Daneshgar, F., Low, G., and Beydoun, G., 2016. Cloud migration process-a survey, evaluation framework, and open challenges. In *Journal of Systems and Software, vol. 120, no. C, pp. 31–69, Oct. 2016*.

Hilton, M., Christi, A., Dig, D., Moskal, M., Burckhardt, S., and Tillmann, N., 2014. Refactoring local to cloud data types for mobile apps. In *Proceedings of the 1st International Conference on Mobile Software Engineering and Systems, ser. MOBILESoft 2014, ACM, 2014, pp. 83–92*.

Hermenier, F., Lorca, X., Menaud, J.-M., Muller, G., and Lawall, J., 2009. Entropy: A consolidation manager for clusters. In Proceedings of the 2009 ACM SIGPLAN/SIGOPS *International Conference on Virtual Execution Environments, ser. VEE '09, ACM, 2009, pp. 41–50.*

Inese, S. Inese, P., Solvita, B., Janis, G., Egils, M., and Edgars, O., 2015. Decomposition of enterprise application: A Systematic literature review and research outlook. In *Information Technology and Management Science, vol. 18, no. 1, pp. 30–36, 2015.*

Inzinger, C., Hummer, W., Satzger, B., Leitner, P., and Dustdar, S.,2014. Generic event-based monitoring and adaptation methodology for heterogeneous distributed systems. In *Software: Practice and Experience, vol. 44, no. 7, pp. 805–822, 2014*.

Inzinger, C., Satzger, B., Leitner, P., Hummer, W., and S. Dustdar, 2013. Model-based adaptation of cloud computing applications. In *Proceedings of the 1st International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2013), 2013, pp. 351–355*.

Jamshidi, P. Pahl, C., and Mendonça, N. C., 2016. Pattern-based multi-cloud architecture migration. In *Software: Practice and Experience, 2016*.

Jennings, B. and Stadler, R, 2015. Resource management in clouds: Survey and research challenges. *In Journal of Network and Systems Management, vol. 23, no. 3, pp. 567–619, Jul. 2015*.

Keras, 2018. Keras: The Python Deep Learning library. Available online: https://keras.io/

Koto, A., Yamada, H., Ohmura, K., Kono, K., 2012. Towards unobtrusive VM live migration for cloud computing platforms. In *Proceedings of the Asia-Pacific Workshop on Systems: ACM; 2012. p. 7.*

Kwon, Y.-W., and Tilevich, E.,2014. Cloud refactoring: Automated transitioning to cloud-based services. In *Automated Software Eng., vol. 21, no. 3, pp. 345–372, Sep. 2014*.

Kyriazis, D, 2013. Cloud computing service level agreements-exploitation of research results. In *Technical report European Commission*. Available online: http://ec.europa.eu/information_society/news room/cf/dae/document.cfm?doc_id=2496, 2013.

Lorido-Botran, T., Miguel-Alonso, J., and Lozano, J. A., 2014. A review of auto-scaling techniques for elastic applications in cloud environments. In *Journal of Grid Computing, vol.12, no.4, pp.559–592, Dec.2014.*

Michael, M. , Moreira, J. E., Shiloach D., and Wisniewski, R. W.,2007. Scale-up x Scale-out: A Case Study using Nutch/Lucene. In *2007 IEEE International Parallel and Distributed Processing Symposium*, Long Beach, CA, 2007, pp. 1-8. doi: 10.1109/IPDPS.2007.370631.

Nguyen, H., Shen, Z., Gu, X., Subbiah, S. and Wilkes, J.,2013. Agile: Elastic distributed resource scaling for infrastructure-as-a-service. In *Proceedings of the 10th International Conference on Autonomic Computing (ICAC 13), USENIX, 2013*.

OASIS, 2017. Topology and orchestration specification for cloud applications version 1.2, Committee Specification Draft 01, OASIS Standard. Available online: http://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.2/csd01/TOSCA-Simple-Profile-YAML-v1.2-csd01.pdf.

OCCI, 2018. Open cloud computing interface, http://occi-wg.org/.

Patiniotakis, I., Papageorgiou, N., Verginadis, Y., Apostolou, D., and Mentzas, G.,2013a. A framework for situation-aware adaptation of service-based applications. In *Adaptive Web Services for Modular and Reusable Software Development: Tactics and Solution. IGI Global, 2013, pp. 253–262*.

Patiniotakis, I., Papageorgiou, N., Verginadis, Y., Apostolou, D., and Mentzas, G., 2013b. Dynamic event subscriptions in distributed event based architectures. In *Expert Systems with Applications , vol. 40, no. 6, pp. 1935–1946, May 2013*.

Qu, C., Calheiros R.N., and Buyya, R., 2016. Auto-scaling web applications in clouds: A taxonomy and survey. In *CoRR, vol. abs/1609.09224, 2016*.

R, 2018. The R project for statistical computing. Available online: https://www.r-project.org/

Ranjan, R., Benatallah, B., Dustdar, S., and Papazoglou, M. P., 2015. Cloud resource orchestration programming: Overview, issues, and directions. *In IEEE Internet Computing, vol. 19, no. 5, pp. 46–56, Sep. 2015.*

Rightscale, 2018. What is auto scaling?. Available online: http://docs.rightscale.com/faq/What_is_auto-scaling.html.

Siddhi,2018. WSO2 Siddhi. Available online: https://github.com/wso2/siddhi.

TensorFlow,2018. An open source library for machine intelligence. Available online: https://www.tensorflow.org/

Vasconcelos, M., Mendonça, N.C., and Maia, P.H. M.,2015. Cloud detours: A non-intrusive approach for automatic software adaptation to the cloud. In Service Oriented and Cloud Computing: *4th European Conference, ESOCC 2015, Taormina, Italy, September 15-17, 2015, Proceedings. Springer, 2015, pp. 181–195*.

Youseff, L., Butrico, M. and Silva, D. D., 2008. Towards a unified ontology of cloud computing. In *Proceedings of the IEEE Grid Computing Environments Workshop (GCE08), 2008*.