Orchestrating BigData Analysis Workflows

Rajiv Ranjan,
Newcastle University

Saurabh Garg
University of Tasmania

Ali Reza Khoskbar
Australian National University

Ellis Solaiman, Philip James
Newcastle University

Dimitrios Georgakopoulos
Swinburne University of Technology

Data analytics has become not only an essential part of day-to-day decision making, but also reinforces long-term strategic decisions. Whether it is real-time fraud detection, resource management, tracking and prevention of disease outbreak, natural disaster management or intelligent traffic management, the extraction and exploitation of insightful information from unparalleled quantities of data (BigData) is now a fundamental part of all decision making processes. Success in making smart decisions by analyzing BigData is possible due to the availability of improved analytical capabilities, increased access to different data sources, and cheaper and improved computing power in the form of cloud computing. However, BigData analysis is far more complicated than the perception created by the recent publicity. For example, one of the myths is that BigData analysis is driven purely by the innovation of new data mining and machine learning algorithms.

While innovation of new data mining and machine learning algorithms is critical, this is only one aspect of producing BigData analysis solutions. Just like many other software solutions, BigData analysis solutions are not monolithic pieces of software that are developed specifically for every application. Instead, they often combine and reuse existing trusted software components that perform necessary data analysis steps. Furthermore, in order to deal with the large variety, volume and velocity of BigData, they need to take advantage of the elasticity of cloud and edge datacentrer computation and storage resources as needed to meet the requirements of their owners. More specifically, many BigData analysis solutions today are organised as data-driven workflows that combine existing and new data analysis steps (which we often refer to as workflow activities).

The flow of information between the analysis activities in a BigData analysis workflow is *dynamic*, meaning it is either determined by the data produced in earlier steps in the workflow (we refer to these as data flow dependencies) or by the structure of the BigData analysis solution that orchestrates the data analysis activities in the workflow (we refer to such structural orchestrations as control flow dependencies). Another dynamic aspect of BigData analytics workflows is mapping data analysis steps/activities to the variety of computing and storage resources of the cloud and edge data center(s) with changing performance. Dealing with these dynamic aspects become more challenging in BigData analysis applications which need to support owner's de-

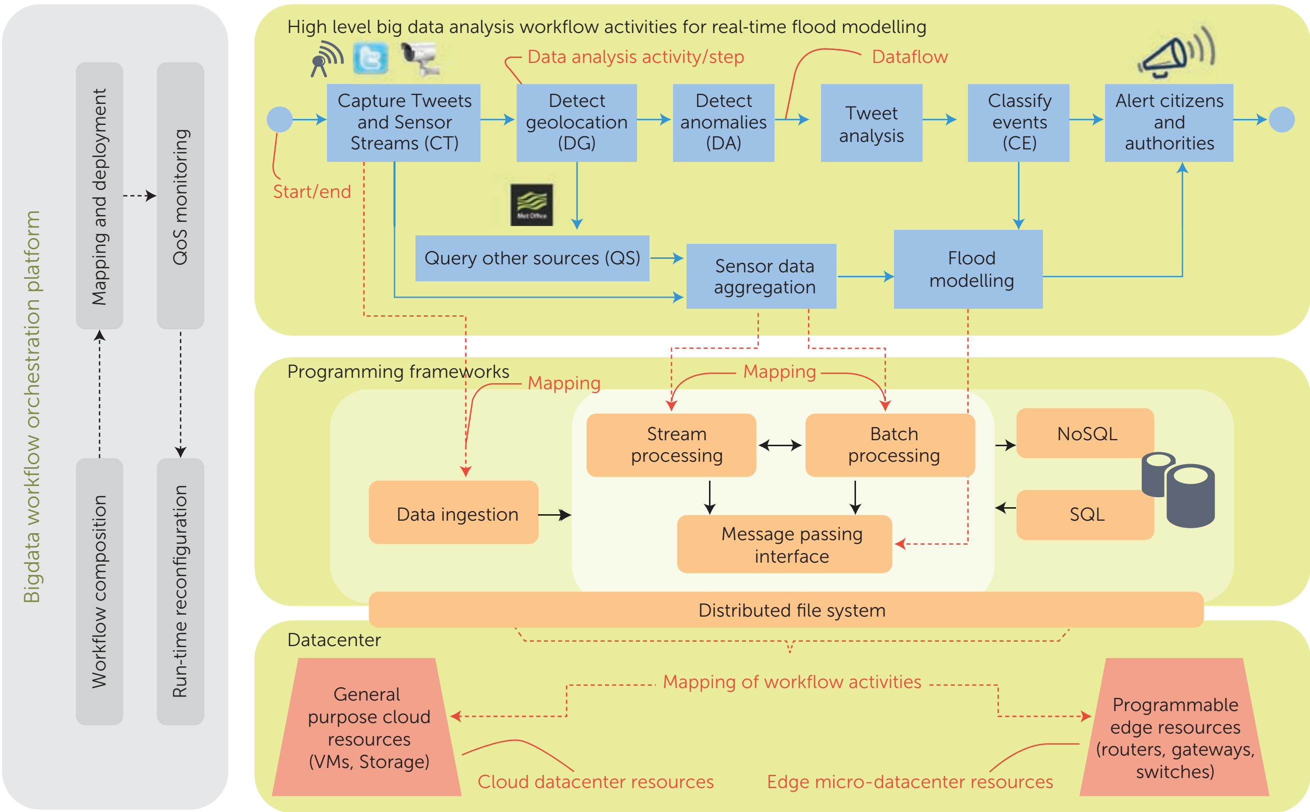

FIGURE 1. Mapping of high level workflow activities of Real-Time Flood Modelling application to programming frameworks and cloud datacenter and/or Edge resources. The workflow orchestration is a cross-cutting issue as it spans across all the layers (analysis activities, programming framework, and datacenters).

cision making requirements (specified in form of Service Level Agreements (SLA)) in real-time. Any delay in meeting their requirements can cause loss of life (such as in disaster prediction and response situations), money (for example, in banking security and fraud situations), or the environment (for instance, in resource exploration). These are some of the real penalties for failing to meet the real-time data analysis requirements in such decision support applications. Computing infrastructures supported by cloud and edge resources can help in solving such problems to some degree by providing elastic and on-demand computing infrastructure. They can also create additional challenges due to the heterogeneous nature of different cloud and edge resources and the dynamically changing performance of their computing infrastructure.

In this Blue Skies installment, we point out the requirement of orchestration systems that can assist in management and execution of such BigData analysis workflows on a cloud and edge infrastructure. We also discuss current state of art and point out open issues in a later section before concluding the article.

An example BigData analysis Workflow

As an example of a BigData analysis workflow, consider Real-Time Flood Modelling (RTFM) for detecting and predicting a flooding event by analyzing tweets and sensor data, as depicted in Figure 1. The RTFM workflow is triggered from long-range forecasting (for example from UK Met Office DataPoint) and radar scans at multiple scales are initiated and passed to statistical processing models, updating probability based forecasts.

As an event progresses, streaming data sources (such as Twitter and ancillary data comparable to traffic flows) can be processed to improve modelling forecasts of a rainfall event's path and intensity. Flood modelling ensembles must then be triggered and matched to known observations (for example from CCTV analysis or rain gauges) in a dynamic system. Flood model outputs are only part of the modelling process providing input into risk and impact models. All of this is happening within a fluid, dynamic, evolving ecosystem where models are refined, re-run or abandoned as new information becomes available. In other words, the workflow includes several top-level data analytics activities. These include long-range forecasting, sensor data aggregation, Tweet analysis, flood modelling, CCTV image processing, and so on. Moreover, the execution of these activities need to be seamlessly coordinated such that real-time decision-making performance objectives (for instance, minimise event detection delay) are constantly achieved under various types of uncertainties (for example, changing data volume and velocity). Hence, the key to seamless execution of this new class of workflows is the issue of resource and data orchestration, which is quite complex due to complex BigData flow pattern and the plethora of BigData programming frameworks, computational models, and infrastructure types (such as cloud datacenters and edge resources) involved:

1. The latency sensitive CCTV image processing activity can benefit by performing "edge analytics" on the video frames by exploiting the on-board processor (edge resources) supported by current generation of CCTV cameras (such as Waggle platform). Using edge analytics techniques has multiple benefits: (i) reduced network congestion achieved by filtering non-relevant events at the edge; and (iii) reduction in event detection latency (for example, detecting dangerous water flow level by analysing real-time images on-board processors available within CCTV cameras) as sensors no longer need to send data to far off cloud datacenters.
2. The flood modelling activity, which does risk analysis by executing a complex hydrodynamic computational model in a message passing interface data programming framework (OpenMPI), should be mapped to the cloud resources, because it is demanding of both storage (due to large historical rainfall records and ensemble city models) and computation (for simulating floods along large river reaches).
3. Workflow activities are inter-dependent and changes in execution characteristics of one activity (at run-time) will influence others. For example, the step handling the flood modelling is dependent on input (on rain and water level thresholds) from the sensor aggregation activity (analysing data from diverse real-time sensors).
4. Tweet analysis activity requires distinct computational models for anomaly detection (flood disasters are anomalous tweets), clustering to combine all the information from different tweets reporting flooded properties in a specific location, and classification to identify major events such as a flood. Moreover, these computational models require either a batch processing or stream processing data programming framework, depending on data characteristics (historical vs. real-time tweets). The activity needs to utilise specialised main memory NoSQL BigData framework and solid state storage resources available in the cloud datacenter to deal with Twitter's data velocity and volume.

To handle these complexities, the underlying Orchestration[1] platform and techniques should be able to dynamically manage a workflow of activities (initially composed based on Domain expert inputs) on the resources available in the cloud datacenter (for example, Amazon Web Services) and on the edge (such as the Waggle platform) driven by processing needs (for instance latency sensitive vs. non-latency sensitive), performance objectives (for example, minimise sensor stream processing latency vs. minimise flood model execution delay) and type of analytic tasks (CCTV image processing vs. flood modelling) relevant to activities. Current BigData workflow orchestration platforms (such as, Apache YARN, Apache Mesos, AWS Lambda, AWS IoT, Google Cloud Dataflow, Google TensorFlow) and research assume either monolithic and purpose-built data analysis solutions that do not need to meet real-time decision support requirements (that is, no workflows, no dynamic orchestration of existing and new data analysis activities, no implementation that can exploit both cloud datacenter and edge resources, and no dynamic tuning of such implementations to meet the users' real-time decision making requirements), or considers only solutions consisting of data analysis workflows that have predictable performance (that is existing orchestration research ignores the complexities of resource and BigData management across cloud datacenter and edge resources for data analytics workflows and does not deal with meeting real-time performance objectives as determined by owner's SLA requirements).

Last but not the least, the existing workflow composition frameworks such as OASIS TOSCA,[2] was developed for web services based workflows and allows workflow modelling and deployment specification up to two levels, software components and cloud services (that is, infrastructure). They do not allow composition of workflows at three different layers (see Figure 1) first at analytical activities, then at programming framework, and finally at datacenter layer, nor do they allow integration of dynamic QoS requirements of decision makers.

Hence, the key research challenges that we perceive are the development of orchestration platforms and techniques that can aid in dynamically composing workflows through an analytical workflow composition framework and developing a robust run-time algorithms that can automatically manage the allocation of the datacenter and edge resources to the analytic activities in response to unexpected changes in data volume, data velocity or other infrastructure level issues (for example, congestion, availability, load-balancing, or anomalies, and so on.).

Understanding the BigData workflow Orchestration Challeneges

To support such complicated and dynamically configurable BigData workflow ecosystems, we need a new orchestration platforms and techniques for managing three layers: (i) sequence of data analysis activities (the workflow) that needs to deal with real-time and historical datasets produced by different sources; (ii) heterogeneous BigData programming frameworks; and (iii) the heterogeneous cloud and/or edge resources. The BigData workflow orchestration is a multi-level resource management and coordination process that spans across workflow activities, BigData programming frameworks and cloud/edge resources. It includes a range of programming operations, from workflow composition, mapping of workflow activities to BigData programming frameworks and cloud/edge resources, to monitoring their end-to-end run-time QoS and SLA statistics (for example, event detection delay, alert delay, load, availability, throughput, utilization, latency, etc.) for ensuring consistency and adaptive management. Briefly stated, major research challenges involved with developing orchestration platforms and techniques for BigData workflow applications include:

Workflow composition: In a BigData analysis workflow (such as RTFM in Figure 1), workloads (data volume and velocity) pertaining to different activities are dependent on each other and changes in execution and data flow of one activity will influence others. For example, the flood modelling activity is dependent on the real-time input on rain and water level thresholds from the sensor data aggregation and CCTV image processing activities. Hence, the hard challenges exist in developing workflow composition framework that can guide the domain experts (for example, flood modeller in a city council office) in specifying, understanding and managing the whole pipeline of activities, data and control flow inter-dependencies and their QoS and/or SLA objectives and measures. For example, suppose we have two owners and/or decision makers for the workflow in Figure 1. The first owner is from a national disaster centre who is interested in information about any infrastructure damage, while another owner from the emergency management services (EMS) may be interested in information about human fatalities and injuries. In this case, the workflow in Figure 1 will dynamically need to compose different clustering activities (infrastructure damages vs human fatalities) that will both utilise the data flow from the anomaly detection activity. Hence, based on decision maker goal workflow composition pattern changes. Moreover, the problem is further complicated by the fact that type and mix of workflow activities, data and control flow inter-dependencies and their QoS and/or SLA measures varies significantly across different application domains (such as, real-time air pollution monitoring, real-time traffic congestion monitoring, remote patient monitoring, etc.).

Workflow mapping: Mapping BigData workflow (graph of data analysis activities) to BigData programming frameworks and cloud/edge resources demands selecting bespoke configurations from abundance of possibilities. Therefore, the mapping process for has to take into account diverse configuration selection decision. For example,

- *BigData programming frameworks:* Select optimal configurations for each framework (for example, in context of stream processing engine such as Apache Storm one needs to determine optimal mix and number of spouts, bolts, and worker instances to minimize data processing latency of stream processing activities)
- *Cloud resources:* Consider configurations such as datacenter location, pricing policy, server hardware features, virtualization features, upstream/downstream network latency, a
- *Edge resources:* Consider configurations such as Edge device (Raspberry Pi 3, UDOO board, esp8266) hardware features (for example, CPU power, main memory size, storage size) , upstream/downstream network latency, supported virtualization features, and so on. Above diverse configuration space coupled with conflicting (trade-off) QoS and SLA requirements leads to exponential growth of potential search space. At the mapping stage, orchestration platform needs to utilise scheduling resource allocation techniques that can allow selection of optimal platform (BigData frameworks) and infrastructure (cloud or edge) configurations for given different workflow components. These techniques also need to consider QoS or SLA requirements such as deployment costs, response time, data processing speed, security level specified by decision makers depending on the application context. These constraints make the mapping problem of each workflow activity to BigData programming framework and datacenter layers NP-Complete. The mapping problem can be easily deducted toto a 0-1 Knapsack or bin-packing problem depending on the constraints given by the decision maker and/or owner.

Workflow QoS monitoring: After the deployment of BigData workflow applications it is important to monitor the run-time QoS and data flow across each activity in the graph, so that administrators and developers can track how application is performing. Much of the difficulty in QoS monitoring from the inherent scale and complexity of BigData workflow application. The problem is complicated because QoS metrics for workflow activities, BigData frameworks, and cloud/edge resources, are not necessarily the same. For example, key QoS metrics are i) event detection and decision making delay for sensor data analysis activity; ii) tweet classification delay and accuracy for Tweet Analysis activity; iii) throughput and latency in distributed data ingestion frameworks (Apache Kafka), iii) response time in batch processing frameworks (Apache Hadoop), (iv) read/write latency and throughput for distributed file system frameworks (for instance, Hadoop Distributed File system); v) server utilization, throughput, and energy-efficiency for cloud resources; and (vi) network stability, throughput optimality, routing delays, fairness in resource sharing, available bandwidth, etc. for the Edge resources.

Therefore it is not clear how i) these QoS metrics could be defined and formulated coherently across workflow activities, BigData programming frameworks, and/or cloud/edge resources and ii) the various QoS metrics should be combined to give a holistic view of data analysis flows. Moreover, to ensure workflow-level performance SLAs we must also monitor workload input metrics (data volume, data velocity, data variety and sources, types and mix of analytics queries) across diverse workflow activities.

Workflow dynamic reconfiguration: The dynamic reconfiguration of BigData workflows in the complex computing infrastructure (Cloud + Edge + multiple BigData frameworks) is complex research problem due to following run-time QoS prediction modelling uncertainties: 1) it is difficult to estimate activity-specific data flow behaviours in terms of data volume to be analysed, data velocity, data processing time distributions, and I/O system behaviour and 2) without knowing the run-time changes to the flow it is difficult to make decisions about the configuration of BigData programming frameworks, cloud and edge resources to be orchestrated so that QoS targets across activities and workflow as whole are constantly achieved; 3) it is difficult to detect causes of QoS anomalies across the complex computing infrastructure due to heterogeneous data flow and QoS measures across multiple workflow activities and the availability, load, and throughput of cloud and/or edge resources can vary unpredictably due to failure or congestion of network links. For example, in Figure 1, velocity of flooding related tweets can increase or decrease based on extent severity of the monsoon. Similarly, during rain gauge sensors can be instrumented to transmit information at much higher velocity and volume during monsoon.

Current State of the art

In this section, we will discuss the current state of the art with respect to the four orchestration challenges in terms of workflow composition, mapping, QoS monitoring, and dynamic reconfigurationto understand to what degree they are able to meet the new end-to-end QoS and SLA requirements of BigData workflow applications.

> After the deployment of BigData workflow applications it is important to monitor the run-time QoS and data flow across each activity in the graph.

Workflow composition: Existing orchestation platform such as Apache Oozie and Linkedin Azkaban supports composition of workflows, which can include multiple batch processing activities hence, does not suit the composition needs of complex workflows such as RTFM (see Figure 1) and others. On the other hand, platforms such as Apache YARN, Apache Mesos, Amazon IoT and Google Cloud Dataflow can support script-based composition of heterogeneous analytic activities on cloud datacenter resources cannot deal with Edge resources. Another example of applying analytical techniques for composing BigData applications is the performance analysis of QoS models based on queuing networks and stochastic Petri nets as mentioned by Ardagna and colleagues.[3] Other works aimed at analysing the Map Reduce paradigm using stochastic Petri nets as well as process algebras and Markov chains are.[4,5] Development like these tend to be greatly focused on a single programming paradigm, in this case Map Reduce (batch processing), and are therefore cannot be easily extended to multiple BigData programming frameworks and heterogeneous computing environments (Cloud + Edge). Workflow modelling and deployment specification frameworks and languages such as TOSCA,[2] OPENSTACK Heat, AWS Cloud Formation template and WS-CDL[6] can assist in web services based workflows for software components and Cloud service. However, BigData workflows are quite complex as each analytical activity itself is a workflow in itself. Moreover, to support decision making process, workflow specification should integrate contextual information, which can be dynamically edited by decision maker.

Workflow mapping: Existing BigData workflow orchestration platforms (Apache YARN, Mesos, and Apache Spark) are designed for homogeneous clusters of cloud resources (agnostic to Edge resources). These orchestrators expect workflow administrators to determine the number and configuration of allocated cloud resource types and provide appropriate software-level configuration parameters for each BigData programing frameworks to which one or more analytic activities are mapped to. Branded price calculators are available from public cloud providers (Amazon, Azure) and academic projects (Cloudrado), which allow comparison of cloud resource leasing costs. However, these calculators cannot recommend or compare configurations across BigData processing frameworks driven diverse QoS measures across workflow activities. In a narrow domain, recent efforts[7-10] have attempted to automate the configuration selection of Hadoop frameworks (batch processing) over heterogeneous cloud-based virtualized hardware resources. Multiple approaches[11] have applied optimization and performance measurement techniques for mapping web applications to cloud by selecting optimal virtual machine configuration

(CPU Speed, RAM Size, cloud location, etc.) based on diverse QoS requirements (throughput, availability, cost, reputation, etc.). However, the configuration space, QoS, and SLA requirements for mapping workflow activities to BigData programming frameworks and cloud/edge resources is fundamentally different from selecting virtual machine configuration for web applications.

Workflow QoS monitoring: BigData Cluster-wide monitoring frameworks (Nagios, Ganglia, Apache Chukwa, Sematex, DMon, SequenceIQ) provide information about QoS metrics (cluster utilization, CPU utilization, memory utilization and nature of application: disk-, network-, or CPU-bound) of virtualized resources that may belong to public or private cloud. These monitoring frameworks[12] do not support workflow activity-level QoS metrics and/or SLAs, which is essential for BigData workflows where change in processing capability of one analytical activity can affect all the activities in the downstream. In the public cloud computing space, monitoring frameworks (Amazon CloudWatch used by Amazon Elastic Map Reduce) typically monitor cloud (agnostic to Edge) VM resource as a black box, and so cannot monitor activity-level QoS metrics and/or data flow. Techniques presented by Alhamazani and colleagues[13] and frameworks such as Monitis[14] and Nimsoft[15] can monitor QoS metrics of web applications hosted on the cloud. Complex event processing and content-based routing applications hosted on clouds. In summary, none of the existing QoS monitoring frameworks and techniques can (i) monitor and integrate data (workload input and performance metrics, disruptive events, SLAs at the platform level, SLAs at the infrastructure) across each activity of the workflow running on multiple BigData processing frameworks and underlying hardware (Cloud + Edge) resources or (ii) detect root causes of workflow activity-level SLA violations and failures across the multiple BigData processing frameworks and hardware resources based on data flow and QoS metrics logs.

Workflow dynamic reconfiguration: Current generation BigData orchestration platforms (YARN, Mesos, Amazon EMR) offer no guarantees about handling failures at workflow-level and/or resource level, nor can they automatically scale or de-scale the platform in response to changes in data volume, velocity or variety, or query types, which can affect the resource requirements of activities within a BigData workflow. There are very few current research works that are trying to address the automatic scaling of single BigData processing framework, batch processing[16] and stream processing.[17] Database community have mostly worked on optimising the query execution performance considering both interleaved18,19 and parallel executions[20,21] via both black-box approaches such online and offline machine learning and white-box approaches for analytical modelling of SQL and/or NoSQL BigData processing frameworks. Existing orchestrators in cloud community that can do online or dynamic reconfiguration have been built specifically for interactive multi-tier web applications.[4,5] However, most of the techniques utilised by them cannot be directly applied to predict data flow metrics (data volume, data velocity, stream operator processing time distributions, query types) or workflow activity-specific QoS metrics (batch processing response time, stream processing latency, data ingestion latency, Tweet analysis accuracy) as BigData workflows are fundamentally different from multi-tier web applications. To make dynamic reconfiguration in the execution of BigData workflow applications, their run-time resource requirements and data flow changes needs to be predicted including any possible failure occurrence. These requirements need to be computed based on inter and intra dataflow of the workflows but also on the user's contextual requirements.

As the concluding remark, current BigData analysis tools and workflow management orchestrators have to evolve to great degree before they can support the requirements of domain-specific BigData workflow applications. Most of these workflows applications are not just monolithic solution but a complex interaction of several BigData programming frameworks, multiple data sources, and heterogeneous Cloud/Edge resources. Each of these applications need to orchestrated to support real time requirements of decision makers expressed in terms of Service Level Agreements.

No prior work has developed workload and resource performance models to enable contention-free scaling and de-scaling of BigData processing frameworks and hardware (Cloud+Edge) resources. In other words, there is no support for new generationBigData workflows' requirements particularly for time-sensitive ones (that is, no workflows, no dynamic orchestration of existing and new data analysis steps, no (Cloud+Edge)-based implementation, and no dynamic tuning of such implementations to meet the owner's decision making requirements), or considers only solutions consisting of data analysis workflows that have predictable performance, which is assumed to be sufficient for its owners (that is, existing research ignores the complexities of cloud and edge resource management for data analysis workflows and does not deal with meeting performance targets as determined by owner's requirements).

Therefore, it is essential that future research consider (1) BigData workflow analysis solutions based on data-driven workflows, (2) mapping such workflows to BigData programming frameworks and Cloud/Edge resources, and (3) manage such mappings and resources to meet specific owner's requirements (or contexts). More specifically, the research community must aim to design new frameworks and novel platforms and techniques that enable decision making by allowing the orchestration of their execution in a seamless manner allowing dynamic resource reconfiguration at runtime.

Acknowledgement

The authors would like thank Mazin Yousif (EIC, *IEEE Cloud Computing*) and Ivona Brandic (TU Wien, Austria) for their valuable feedback, which significantly contributed to improving quality of this column.

References

1. R. Ranjan, B. Benatallah, S. Dustdar and M. P. Papazoglou, "Cloud Resource Orchestration Programming: Overview, Issues, and Directions," *IEEE Internet Computing*, vol. 19, no. 5, 2015, pp. 46–56; doi: 10.1109/MIC.2015.20.
2. Mendling, J., & Hafner, M. (2005). From Inter-organizational Workflows to Process Execution: Generating BPEL from WS-CDL. *On the Move to Meaningful Internet Systems 2005: OTM 2005 Workshop*, Springer, 2005.
3. D. Ardagna et al., "Modeling Performance of Hadoop Applications: A Journey from Queueing Networks to Stochastic Well Formed Nets," *Algorithms and Architectures for Parallel Processing - 16th International Conf.* (ICA3PP 16), 2016, pp. 599–613.
4. A. Castiglione et al., "Exploiting mean field analysis to model performances of big data architectures," *Future Generation Comp. Syst.*, vol. 37, 2014, pp.203–211.
5. D Bruneo et al., Analytical Modeling of Reactive Autonomic Management Techniques in IaaS Clouds. *Proc. 2015 IEEE 8th International Conf. on Cloud Computing* (CLOUD 15), 2015, pp. 797–804.
6. T. Binz et al., "Portable Cloud Services Using Tosca," IEEE Internet Computing, vol. 16, no. 3, pp. 80–85.
7. H. Herodotou et al., "A What-if Engine for Cost-based Mapreduce Optimisation," *IEEE Data Engineering Bulletin*, vol. 36, no. 1, 2013, pp. 5–14.
8. G. Lee, B. Chun, and H. K. Randy, "Heterogeneity-Aware Resource Allocation and Scheduling in the Cloud," *Proc. 3rd USENIX Conf. on Hot topics in cloud computing* (HotCloud 11), 2011, p. 4.
9. P. Lama and X. Zhou, "AROMA: Automated Resource Allocation and Configuration of MapReduce Environment in the Cloud," *Proc. 9th International Conference on Autonomic Computing* (ICAC 12), 2012, pp. 63–72.
10. K. Kambatla et al., "Towards Optimising Hadoop Provisioning in the Cloud," *Proc. 2009 Conf. on Hot topics in cloud computing* (HotCloud 09), 2019, Article 22.
11. R. Ranjan, J. Kolodziej, L. Wang and A. Y. Zomaya, "Cross-Layer Cloud Resource Configuration Selection in the Big Data Era," *IEEE Cloud Computing*, vol. 2, no. 3, 2015, pp. 16–22;. doi: 10.1109/MCC.2015.64
12. K. Alhamazani, R. Ranjan et al., "An Overview of the Commercial Cloud Monitoring Tools: research dimensions, design issues, and state-of-the-art," *Springer Journal of Computing*; doi: 10.1007/s00607-014-0398-5.
13. K. Alhamazani, R. Ranjan et al., "CLAMS: Cross-Layer Multi-Cloud Application Monitoring-as-a-Service Framework," *Proc. of the 11th IEEE Int. Conf. on Services Computing* (SCC 14), 2014 pp. 283–290.

14. TeamViewer, "Monitis," Feb. 2015; http://portal.monitis.com/.
15. CA Technologies "Proactively enure experience of cloud and hybrid IT infrastructures," Feb. 2015; https://www.ca.com/us/products/ca-unified-infrastructure-management.html
16. A. Abouzeid et al., "Hadoopdb: An Architectural Hybrid of Mapreduce and DBMS technologies for Analytical Workloads," *Proc. of the VLDB Endowment*, (PVLDB 09), 2009, pp. 925–933, 2009.
17. R.CastroFernandez, et al., "Integrating Scale Out and Fault Tolerance in Stream Processing using Operator State Management," *Proc. of the International Conf. on Management of Data* (SIGMOD 13), 2013, pp. 725–736.
18. A. Ganapathi et al., "Predicting Multiple Metrics for Queries: Better Decisions Enabled by Machine Learning", *Proc. of the IEEE International Conference on Data Engineering* (ICDE 2009), 2009, pp. 592-603.
19. M. Akdere et al., "Learning-based Query Performance Modeling and Prediction," *Proc. of the 2012 IEEE International Conf. on Date Engineering* ICDE 2012, 2012, pp. 395–401.
20. J. Duggan et al.,"Performance Prediction for Concurrent Database Workloads," *Proc. Of the International Conf. on Management of Data* (SIGMOD 11), 2011, pp. 337–348.
21. W. Wu et al., "Towards Predicting Query Execution Time for Concurrent and Dynamic Database Workloads," *Proc. of the VLDB Endowment* (PVLDB 12), 2012, pp. 925–936.