POLITECNICO DI MILANO

Dipartimento di Elettronica Informazione e Bioingegneria
Doctoral Program in Information Technology

Performance Models, Design and Run Time Management of Big Data Applications

Doctoral Dissertation by:
Eugenio Gianniti

Supervisor:
Prof. Danilo Ardagna
Co-Advisor:
Dott. Michele Ciavotta
Co-Advisor:
Dott. Marco Lattuada
Tutor:
Prof. Luciano Baresi
The Chair of the Doctoral Program:
Prof. Andrea Bonarini

2018 — XXX Cycle

Abstract

Nowadays the big data paradigm is consolidating its central position in the industry, as well as in society at large. Lots of applications, across disparate domains, operate on huge amounts of data and offer great advantages both for business and research. As data intensive applications (DIAs) gain more and more importance over time, it is fundamental for developers and maintainers to have the support of tools that enhance their efforts since early design stages and until run time. The present dissertation takes this perspective and addresses some pivotal issues with a quantitative approach, particularly in terms of deadline guarantees to ensure quality of service (QoS).

Technically interesting scenarios, such as cloud deployments supporting a mix of heterogeneous applications, pose a series of challenges when it comes to predicting performance and exploiting this information for optimal design and management. Performance models, with their potential for what if analyses and informed design choices about DIAs, can be a major tool for both users and providers, yet they bring about a trade-off between accuracy and efficiency that may be tough to generally address. The picture is further complicated by the adoption of the cloud technology, which means that assessing operating costs in advance becomes harder, but also that the contention observed in data centers strongly affects big data applications' behavior. For all these reasons, ensuring QoS for novel DIAs is a difficult task that needs to be addressed in order to favor further development of the field.

Over this background, the present dissertation takes two main routes towards facing such challenges. At first we describe and discuss a number of performance models based on various formalisms and techniques. Among these, there are both basic models aimed at predicting specific metrics, like response time or throughput, and more specialized extensions that target the impact on big data systems of some design decisions, e.g., privacy preserving mechanisms or cloud pricing models. On top of this, the proposed models are variously positioned across the spectrum between efficiency and accuracy, thus enabling different trade-offs depending on the main requirements at hand. This is relevant in the second main part of this dissertation, where performance prediction is at the core of some formulations for capacity allocation and cluster management. In order to obtain optimal solutions to these problems, in one case at design time and in the other at run time, we adopt both mathematical programming and several performance models, according to the different constraints on solving times and accuracy.

More in detail, we propose performance models based on queueing networks (QNs), stochastic well formed nets (SWNs), and machine learning (ML). This variety is justified by the different uses of each methodology. ML pro-

vides algebraic formulas for execution times, which are perfectly fit to be added as constraints in our optimization problems' mathematical programming formulations, thus yielding initial solutions in closed form. Since ML can reliably provide accurate predictions only in regions properly explored during the training phase, the optimal solution is searched via a simulation-optimization procedure based on analytical models like QNs or SWNs, which in contrast are quite insensitive to the parameter range of evaluation, being devised from first principles. These kind of models boast relative errors below 10 % on average when predicting response times.

In terms of optimization, first of all we consider the design time problem of capacity allocation in a cloud environment. The design space is explored via both ML and simulation techniques, so as to choose the best virtual machine type in the catalog offered by cloud providers and, subsequently, determine the minimum cost configuration that satisfies QoS constraints. We show also how this optimization approach was applied during the design phase of a tax fraud detection product developed by industrial partners, i.e., NETF Big Blu. Afterwards we also considered the run time issue of finding the minimum tardiness schedule for a set of jobs when the current workload exceeds predictions and the deployed capacity is not enough to ensure the agreed upon QoS. Thanks to the varied efficiency of performance models, it is possible to solve the design time problem in a matter of hours, whilst run time instances are solved within minutes, consistently with the different requirements.

Contents

List of Figures

List of Tables

List of Corrections

CHAPTER 1

Introduction

Many analysts point out that during these years technologies and methodologies that fall within the sphere of big data have swiftly pervaded and revolutionized many sectors of industry and economy, becoming one of the primary facilitators of competitiveness and innovation [63].

IDC reports that big data used to concern highly experimental projects, yet its market is growing from $130 billion in 2016 to $203 billion in 2020, with a compound annual growth rate of 11.9 %, with the banking and manufacturing industries leading in terms of investment [128]. Big data applications offer many business opportunities that stretch across industries, especially to enhance performance, as in the case of recommendation systems. Furthermore, data intensive applications (DIAs) can also help governments in obtaining accurate predictions, for instance quality weather forecasts to prevent natural disasters and ease the development of appropriate policies to improve the population's life quality. To corroborate these considerations, notice that the Obama government announced $200-million worth of investment to boost big data related industries and positioned this strategy into the national agenda in 2012. In addition, big data systems are increasingly exerting a central force on society, thus requiring the development of intelligent systems providing quality of service (QoS) guarantees to their users.

This dissertation reckons DIAs' importance in today's economy and tackles some relevant problems linked to their adoption. Any fruitful approach to the optimization of big data applications must rely on effective performance models, which are basic tools needed for the prediction of execution times or the determination of costs given QoS constraints. For this reason, the initial part of this dissertation discusses and compares a range of performance models, based on various formalisms and techniques. Building on top of these, we also propose novel solutions to optimization problems that play a major role in DIAs' design and operation. In particular, we will detail formulations for the capacity planning problem at design time, as well as for the run time management of workload peaks.

This chapter proceeds as follows. Section 1.1 presents the main challenges to face when dealing with DIAs in the cloud. Later on, Section 1.2 introduces

the most relevant big data frameworks, which were studied in the development of this dissertation, then Section 1.3 similarly introduces convolutional neural networks. After that, Section 1.4 motivates the choice to investigate performance models and summarizes the proposed ones, while Section 1.5 is about the optimization techniques. Section 1.6 acknowledges the international research projects that supported this work and Section 1.7 introduces the main questions that drive the content of this dissertation. The chapter ends with a list of the relevant publications where I contributed and the thesis organization.

1.1 Challenges for QoS Aware Big Data Systems

As DIAs acquire a central position in society, the IT field should shift from simply building systems to developing intelligent systems that provide QoS. Yet, predicting the performance of big data applications in scenarios of technical interest, notably a mix of applications running concurrently in a cloud system, is very challenging. Big data applications are characterized by changing behavior during execution: for instance, they require initially a lot of CPUs, then a lot of network capacity, to later switch between the two, with sometimes complex patterns. Moreover, to cope with the large amount of data, such applications often run in parallel stages. The performance (and thus QoS) of parallel applications is often harder to predict due to synchronization overheads. In summary, the most relevant challenges are:

Performance prediction via models Designing new models to predict the performance of applications, in terms of execution time, given certain resources, is key to providing QoS to application customers. Such models should be accurate and efficient, i.e., provide an estimate quickly. The use of accurate models is beneficial for both cloud providers and end users: for cloud providers, models can trigger run time adaptations to provide QoS guarantees; for end users, they can support what if analysis and enable taking more informed decisions on the resources to use. It is important that the models run reasonably fast, i.e., provide responses quick enough to drive run time adaptations. However, model accuracy and efficiency are two often conflicting objectives. More sophisticated models, capturing in more details different aspects of the application execution, are often more accurate, but also very costly to run. Thus finding the best trade-off between these two goals is a major challenge.

Cost predictability in the cloud Designing new models to estimate the costs in terms of cloud resources to run big data applications is key. An accurate estimate enables more efficient scheduling of resources, including cost-effective utilization of data centers.

Ensuring QoS on a budget Big data applications are supported by cloud infrastructures that, for resource contention, can be affected by performance decline. For this reason, one of the major challenges for big data applications is to define mechanisms and policies that implement resource partitioning and management in a way that cloud data centers' resources are used efficiently, providing differentiated service levels to customers according to the price of the resources.

In relation to the above challenges, at first this dissertation proposes various performance models and compares the advantages each brings to the table. Undoubtedly these models give a strong contribution for the prediction of response times and other performance metrics, but they can also be used to assess DIAs' resource requirements. Moreover, they are at the core of the optimization methods presented as the other important part of this dissertation. The choice among a gamut of alternative modeling techniques is instrumental to face the different issues that arise in the diverse scenarios of interest. Specifically, the adoption of optimization at design or run time is characterized by different constraints, for instance in terms of the time taken to obtain a solution or the required accuracy.

1.2 Big Data Frameworks

One of the pillars on which the big data revolution is based is the MapReduce paradigm, which has allowed for massive scale parallel analytics [76]. MapReduce, a programming model and a scalable and fault tolerant run time environment [35], is the core of Apache Hadoop, open source framework that has proven capable of managing large datasets over either commodity clusters or high performance distributed topologies [130].

The MapReduce framework became the most popular platform for data analytics because of its simplicity, generality, and maturity [138]. A data processing request under the MapReduce framework, called a job, consists of two types of tasks: map and reduce. A map task reads one data chunk and processes it to produce intermediate results, then reduce tasks fetch the partially processed data and carry out further computation to generate the final result [126].

Hadoop is an open source implementation of MapReduce ready for production deployments and used for applications like log file analysis, database (DB) querying, web indexing, report generation, machine learning research, scientific simulation, bioinformatics, and financial analysis [58, 129]. Hadoop's success has been planetary; it attracted the attention of both academia and industry as it overtook the scalability limits of traditional data warehouse and business intelligence solutions [76]. For the first time, processing unprecedented amounts of structured and unstructured data was within reach, thus opening up, suddenly, a whole world of opportunities.

From the technological perspective, MapReduce is capable of analyzing very efficiently large amounts of unstructured data, i.e., it is a viable solution to support both the variety and volume requirements of big data analyses [74]. Cloud platforms make MapReduce an attractive framework for organizations that need to process large datasets, but lack the computing and human resources to install and manage a cluster. Moreover, Hadoop 2.x recently introduced a wide set of performance enhancements, such as SSD support, caching, and I/O barriers mitigation. IDC had estimated that Hadoop touched half of the world data by 2015 [68], supporting both traditional batch and interactive data analysis applications [109]. Paradoxically, the MapReduce paradigm, which has contributed so much to Hadoop's rise, is steadily declining in favor of solutions based on more generic and flexible processing models. Among these, Apache Spark is a framework that is enjoying considerable success and

that, according to analysts, is expected to dominate the market for the next decade [40].

The rigid division between map and reduce requires to subdivide a complex application into a directed acyclic graph (DAG) of MapReduce jobs, comprising tasks that perform a specific computation on partitions/splits of the input data. In this case, the MapReduce paradigm forces to store the results of each intermediate phase on disk, thus being unsuitable for applications requiring a low latency between different phases, along with general application QoS guarantees. Other frameworks, such as Tez [106] and Spark [135], have been introduced to address this problem. Although Tez can handle general DAGs of MapReduce phases, it still requires to write each stage's results on disk. On the other hand, Spark can exploit a set of primitives to request the caching of partial results in memory, thus allowing lower latency and better performance. Spark has been developed on the resilient distributed dataset (RDD) concept [134], a novel distributed memory abstraction providing a restricted form of memory sharing. In practice, Spark can easily obtain a 10x speedup over Hadoop on specific scenarios [135]. This motivates Spark's widespread adoption, which made it the leading framework for data science at scale. Acknowledging its newly acquired importance for big data, the focus of this dissertation shifted from MapReduce, which in this fast paced field can now be considered legacy, to Apache Spark.

1.3 Deep Learning Frameworks

Among DIAs, an important role is played also by neural networks (NNs). Nowadays, convolutional neural networks (CNNs) find application across industries, most notably for image recognition and classification tasks, which represented the first successful adoption of the technique [71]. Ranging from medical diagnosis to public security, deep learning (DL) methods are fruitfully exploited in a wide gamut of products. In addition to the established applications, there is ongoing work on the technique's adaptation for other use cases, like speech recognition [107] and machine translation [17]. Over time, many frameworks have been developed to provide high level APIs for CNN design, learning, and deployment. Among the most well known, we recall Torch,[1] PyTorch,[2] TensorFlow,[3] and Caffe.[4]

Contrasting to big data frameworks, CNNs generally do not process unstructured data, instead networks themselves are somewhat tailored for the intended input data. Adopting the image recognition example to support intuition, CNNs are peculiar in that they consist of two main portions: an ordinary NN acting as classifier, thus distinguishing different image categories, constitutes the final end of the structure, while a convolutional part automatically extracts features to feed into the classifier. In layman's terms, the learning process enables the convolutional layers to recognize high level features such as beaks or paws, which in turn help the classifier in telling cats from birds.

[1] http://torch.ch
[2] http://pytorch.org
[3] https://www.tensorflow.org
[4] http://caffe.berkeleyvision.org

Usually DL models are trained relying on GPGPU systems (even in clusters for experimental environments [127]), which allow to achieve from 5 up to 40x time improvement when compared to CPU deployments [18]. Motivated by the relevance assumed by these applications, in the following we also propose two performance models specifically tailored for CNNs.

1.4 Performance Modeling for DIAs

In spite of all the fuss around big data technologies, it is still undeniably true that fully embracing them is a very complex process. Many efforts have been made to make this technology accessible, but establishing a production ready deployment is time consuming, expensive, and resource intensive. Not to mention the fact that fine tuning is still often perceived as a kind of occult art.

It is widely held that there is a clear need for an *easy button* to accelerate the adoption of big data analytics [54]. That is why many companies have started offering cloud-based big data solutions, like Microsoft HDInsight, Amazon Elastic MapReduce, or Google Cloud Dataproc, while IDC estimates that, by 2020, nearly 40 % of big data analyses will be supported by public clouds [48]. The advantages of this approach are manifold. For instance, it provides an effective and cheap solution for storing huge amounts of data, whereas the pay per use business model allows to cut upfront expenses and reduce cluster management costs. Moreover, the elasticity can be exploited to tailor clusters capable to support DIAs in a cost-efficient fashion. Yet, provisioning workloads in a public cloud environment entails several challenges. In particular, the space of configurations (in particular, in terms of nodes type and number) is very large, thus identifying the exact cluster configuration is a complex task, especially in light of the consideration that the blend of job classes in a specific workload and their resource requirements may also vary over time.

At the very beginning, MapReduce jobs were meant to run on dedicated clusters to support batch analyses via a FIFO scheduler [100, 102]. Nevertheless, DIAs have evolved and nowadays large queries, submitted by different users, need to be performed on shared clusters, possibly with some guarantees on their execution time [139, 140]. This is not a loose requirement, indeed, as one of the major challenges [80, 119] is to predict the application execution times with a sufficient degree of accuracy. In such systems, capacity allocation becomes one of the most important aspects. Determining the optimal number of nodes in a cluster shared among multiple users performing heterogeneous tasks is a relevant and difficult problem [119].

Unfortunately, modeling the performance of such systems is very challenging. Indeed, production Hadoop environments are nowadays very large massively parallel systems where map and reduce tasks coordinate exhibiting precedence constraints and strict synchronization barriers. Additionally, in our context, the stakeholders interested in the performance evaluation of Hadoop processes are its users rather than its developers. Therefore, the complexity and novelty of these systems together with the lack of full knowledge of their development details make unclear the concepts that should be included in a performance model in order for them to be both accurate and manageable by performance evaluation tools.

Moreover, with Hadoop 2, resources are dynamically allocated between

the map and reduce stages. While in early Hadoop versions CPU *slots* and other resources were separated between mappers and reducers using a static approach, in Hadoop 2 *containers* (both for MapReduce and Spark) are distributed among ready tasks in a dynamic fashion by YARN. On the one hand, this allows a better cluster utilization, on the other hand performance modeling became much more difficult. In Spark, cluster resources are scheduled to process part of the operations on RDDs: to obtain an RDD, Spark first builds a DAG with its dependencies, then processes each stage providing a certain amount of resources, based on data locality.

Because of all these reasons, predicting the execution time of Hadoop or Spark jobs is usually done empirically through experimentation, requiring a costly setup [53]. Performance prediction models are extremely useful to aid development and deployment of big data applications, either for design time decisions or run time system reconfiguration. Design time models can help, e.g., to determine the appropriate size of a cluster or to predict the budget required to run Hadoop or Spark in public clouds. Such models can be used also at run time, allowing for a dynamic adjustment of the system configuration [11, 101], e.g., to cope with workload fluctuations or to reduce energy costs.

Analytically modeling DIAs is very challenging due to the great number of parameters that have to be investigated. To ensure analytical model (AM) tractability in such complex systems, AM-based performance models typically rely on simplifying assumptions that reduce their accuracy. On the other hand, machine learning (ML) deals with the study and construction of algorithms that can learn from data and make predictions on it without a priori knowledge about the internals of the target system. In recent years, a growing number of successful researches were done to explore the possibility of using ML techniques to predict the performance of complex computer systems [61, 73, 132]. To be able to predict accurately, the ML model should be built during a training phase with a sufficient amount of experimental data from different workloads, using various parameters and configurations. However, running several experiments in a cloud environment would be costly and time consuming. On top of this, though ML often provides good accuracy in regions for which it is well trained, it shows poor precision in regions for which none or very few samples are known.

Gray box modeling [41, 43, 62] is a new approach for performance modeling and prediction that tries to achieve the best of the AM and ML worlds by mixing the two. Such models can be exploited to support design time decision making during the development and deployment phases of big data applications. These models can then be also kept alive at run time to conduct the dynamic adjustment of the system configuration [101]. In this context, the performance models proposed in the present dissertation span the whole spectrum from AMs to MLs, also with a hybrid approach aimed at attaining similar accuracy despite the use of less operational data, which entails savings on ad hoc experimental deployments.

In spite of the widespread adoption of DL systems, still there are few studies taking a system perspective that aim at investigating how, for example, the training time changes when running on different GPGPUs or by varying the number of training iterations or the batch size [18, 57]. DL applications are characterized by a large number of design choices that often do not ap-

ply readily to other domains or hardware configurations, up to the point that even advanced users with considerable DL expertise fail at identifying optimal configuration settings [57].

1.4.1 Contributions

The originality of this dissertation consists in an array of modeling techniques capable of catching the system behavior under the dynamic assignment of the available cluster resources. We assume that the cluster is governed by the Capacity Scheduler, which partitions the available resources among multiple customers through queues, each queue being regulated by a FIFO policy. Based on this assumption, we devised several performance models relying on different formalisms, so as to evaluate their relative accuracy and efficiency and tailor them to the specific requirements of each use case. In particular, we investigated queueing networks (QNs), stochastic well formed nets (SWNs), fluid Petri nets, discrete event simulators (DESs), as well as ML approaches.

When the focus is on model expressiveness rather than fast prediction, for instance during the design phase, it is possible to estimate DIAs execution times for multiple users and under unreliable resources. In particular, we analyze a cloud-based scenario where the cluster, to save execution costs, includes also spot virtual machines (VMs) [47]. The utilization of spot VMs offers large discounts in VM prices, with the drawback of a non-guaranteed availability level. We combine the performance and availability dynamics of cloud resources in a single *performability* model that allows for evaluating how failures caused by a sudden deallocation of VMs by the cloud providers degrade system performance.

In this dissertation, we also present a method to learn performance models for CNNs running on a single GPGPU. The main metrics under investigation are the forward time, relevant to quantify the time taken for classification when the trained network is deployed, and the gradient computation time, which on the other hand is important during the learning phase.

The validation of all the proposed models has been carried out with experiments on real systems, considering a number of target deployments. Namely, we considered public clouds with Amazon EC2 and Microsoft Azure HDInsight, community clouds with CINECA, the Italian supercomputing center, as well as on premises deployments with an internal installation, based on IBM POWER8 processors, at Politecnico di Milano. In order to make the validation both reproducible and reliable, we chose the TPC-DS benchmark, which is an industry standard for data warehouse and business intelligence applications. Alongside these analyses based on the benchmark, we also considered some case studies proposed by industrial partners, including also ML workloads. The presented simulation models show, on average, a good accuracy with respect to measurements: their mean relative errors are 14.13 % for QNs and 9.08 % for SWNs.

1.5 Capacity Planning and Resource Management Optimization

Given a set of performance modeling techniques, with their pros and cons, the next step of this dissertation is the development of optimization methods to solve problems related to the management of DIAs in the cloud. Specifically, we focused on the issues of capacity planning at design time, while at run time on the rebalancing of the available resources to meet the requirements of newly submitted jobs, possibly going beyond the foreseen workload and associated capacity.

We formulate the design time capacity planning problem by means of a mathematical model, with the aim of minimizing the cost of cloud resources. The problem considers multiple VM types as candidates to support the execution of big data applications from multiple user classes. Cloud providers offer VMs of different capacity and cost. Given the complexity of virtualized systems and the multiple bottleneck switches that occur in executing DIAs, very often the largest available VM is not the best choice from either the performance or performance/cost ratio perspective [53, 139]. Through a search space exploration, our approach seeks the optimal VM type and number of nodes considering also specific cloud provider pricing models (namely, reserved, on demand, and spot instances). The underlying optimization problem is NP-hard and is tackled by a simulation-optimization procedure able to determine an optimized configuration for a cluster managed by the YARN Capacity Scheduler. DIA execution times are estimated by relying on a gamut of models, including ML and simulation based on QNs, stochastic Petri nets (SPNs) [9], as well as an ad hoc simulator, dagSim [3], especially designed for the analysis of applications involving a number of stages linked by DAGs of precedence constraints. This property is common to legacy MapReduce jobs, workloads based on Apache Tez, and Spark-based applications.

Analogously, the run time problem of resource reallocation is formulated as a distinct mathematical programming model whose objective is the minimization of tardiness. In this case the focus shifts to private clouds, where the previously applied pricing model is not relevant. Along the same lines, it is not possible to choose a different VM type, since this decision was already taken during the design phase. Exploiting another simheuristic procedure, it is possible to obtain the optimal reallocation of resources that enables hard deadline DIAs to meet their service level agreements (SLAs) and loosens the constraints on soft deadline jobs, in order for them to achieve the minimum overall weighted tardiness.

1.6 Acknowledgment

This dissertation has been developed within the framework of two H2020 projects: DICE and EUBra-BIGSEA. The former has as main goal a DevOps framework for designing DIAs and exploiting information extracted from their deployments to improve on such design, whilst the latter aims at a run time environment to provide QoS guarantees for big data, both in data centers and in the cloud.

Relying on the hereby presented performance models and optimization

methods, we obtained two main outcomes: D-SPACE4Cloud[5] and several modules in the EUBra-BIGSEA ecosystem. D-SPACE4Cloud is a piece of software designed to help system administrators and operators in the capacity planning of shared big data clusters hosted in the cloud, so as to support both batch and interactive applications with deadline guarantees. We believe that being able to successfully address this problem at design time enables developers and operators to make informed decisions about the technology to use, while also allowing for the full exploitation of the potential offered by the cloud infrastructure. On the other hand, at run time it is no more possible to take far reaching choices, yet the EUBra-BIGSEA architecture enables the optimal management of the available resources, so as to reduce the impact on QoS of unforeseen workload spikes.

1.7 Research Questions

Overall, this dissertation's main contributions revolve around four research questions. These range from the accuracy of performance models and prediction techniques, but also their time efficiency, to the effectiveness of optimization procedures, even when applied in time constrained scenarios. It is also relevant to investigate the impact of different deployment options on performance, with particular attention to the savings enabled by non-obvious interactions among workloads and underlying computational capabilities. In the end, the proposed optimization techniques may allow for design choices that go beyond simple cluster sizing, then such a possibility should be assessed. The following paragraphs expand with more details the above mentioned problematics.

Research question 1. *Which performance models can be applied to DIAs in an accurate and efficient way? In particular, which have fitting characteristics for design time optimization techniques or, alternatively, for run time management of big data deployments?*

Searching for the optimal configurations to deploy DIAs implies the basic requirement of predicting with a fair confidence their performance, depending on the amount and type of allocated computational resources. If design time optimization is less constrained in terms of convergence times, conversely when operating at run time it is fundamental to shrink the time taken for the optimization procedure, so as to keep at a minimum the impact on DIAs execution. Hence, it is relevant to investigate the trade-offs between accuracy and prediction speed enabled by different alternative techniques, thus determining which better fit either application scenario and highlighting the compromises that might possibly be needed.

Research question 2. *How to identify the minimum cost configuration at design time and how to manage it at run time under high load conditions? Are the proposed optimization methods accurate?*

One of the major issues when dealing with complex pieces of software, such as DIAs, is that they provide lots of configuration parameters, a number of which have effects on the overall performance. This problem is further

[5]DICE System Performance and Cost Evaluation for Cloud

worsened by the vast catalog of alternative deployment choices enabled by the cloud. In a similar setting, it is fundamental to devise appropriate methods capable of efficiently exploring the state space, since an exhaustive search would be utterly impossible. Even more so when the goal is the run time management of DIAs clusters, as the more stringent optimization time constraints exacerbate the issue. In the end, applicability of the proposed techniques heavily depends on their accuracy, hence it is important to assess it via an extensive experimental campaign.

Research question 3. *When looking for the minimum cost deployment, are there any dominant configurations? Is it possible, instead, that different workloads lead to different minimum cost configurations? What is the impact of providers' catalogs on these considerations?*

Common practices tend to associate specific classes of instances to applications based on the matching among VMs' computational capabilities and DIAs' requirements. An immediate example is offered by Apache Spark, which requires a large central memory on each worker node, so as to achieve faster processing in iterative applications via the caching of RDDs. Cloud providers often offer computing and memory optimized instances to satisfy such needs. The focus in this research question is on investigating whether such a preliminary choice is always optimal, or if the complex dependencies between frameworks and deployment options enable different minimum cost configurations based on varying concurrency levels, SLAs, and so forth. Moreover, it is interesting to assess whether such behavior can be observed also across various providers.

Research question 4. *Are these techniques useful to investigate application architectural choices at design time?*

Several architectural choices might have an impact on DIAs' observed performance and this, in turn, affects the optimal configuration for what concerns both the type and number of allocated resources. A natural implication is the adoption of the devised methods for modeling and optimization at design time, when it is possible to quantitatively explore several choices and to assess their effect on the final performance and operational costs.

Personal Publications

[8] Danilo Ardagna, Enrico Barbierato, Athanasia Evangelinou, Eugenio Gianniti, Marco Gribaudo, Túlio B. M. Pinto, Anna Guimarães, Ana Paula Couto da Silva, and Jussara M. Almeida. "Performance Prediction of Cloud-Based Big Data Applications." In: *Proceedings of the 2018 ACM/SPEC International Conference on Performance Engineering*. ICPE '18. Berlin, Germany: ACM, 2018, pp. 192–199. ISBN: 978-1-4503-5095-2. DOI: 10.1145/3184407.3184420.

[9] Danilo Ardagna, Simona Bernardi, Eugenio Gianniti, Soroush Karimian Aliabadi, Diego Perez-Palacin, and José Ignacio Requeno. "Modeling Performance of Hadoop Applications: A Journey from Queueing Networks to Stochastic Well Formed Nets." In: *16th International Conference on Algorithms and Architectures for Parallel Processing*. Ed. by Jesús Carretero, Javier García Blas, Ryan K. L. Ko, Peter Mueller, and Koji Nakano. Vol. 10048. Lecture Notes in Computer Science. Springer, Dec. 2016, pp. 599–613. ISBN: 978-3-319-49582-8. DOI: 10.1007/978-3-319-49583-5_47.

[15] Ehsan Ataie, Eugenio Gianniti, Danilo Ardagna, and Ali Movaghar. "A Combined Analytical Modeling Machine Learning Approach for Performance Prediction of MapReduce Jobs in Cloud Environment." In: *SYNASC* (Timisoara, Romania). 2016.

[32] Michele Ciavotta, Eugenio Gianniti, and Danilo Ardagna. "Capacity Allocation for Big Data Applications in the Cloud." In: *ICPE'17 Companion* (L'Aquila, Italy). Apr. 2017, pp. 175–176.

[33] Michele Ciavotta, Eugenio Gianniti, and Danilo Ardagna. "D-SPACE4Cloud: A Design Tool for Big Data Applications." In: *ICA3PP* (Granada, Spain). Lecture Notes in Computer Science 10048. Springer, Dec. 2016, pp. 614–629.

[50] Eugenio Gianniti, Danilo Ardagna, Michele Ciavotta, and Mauro Passacantando. "A Game-Theoretic Approach for Runtime Capacity Allocation in MapReduce." In: *CCGrid* (Madrid, Spain). May 2017, pp. 1080–1089.

[51] Eugenio Gianniti, Alessandro Maria Rizzi, Enrico Barbierato, Marco Gribaudo, and Danilo Ardagna. "Fluid Petri Nets for the Performance Evaluation of MapReduce and Spark Applications." In: *ACM Performance Evaluation Review* 44.4 (Mar. 2017), pp. 23–36. DOI: 10.1145/3092819.3092824.

[52] Eugenio Gianniti, Alessandro Maria Rizzi, Enrico Barbierato, Marco Gribaudo, and Danilo Ardagna. "Fluid Petri Nets for the Performance Evaluation of MapReduce Applications." In: *InfQ* (Taormina, Italy). Oct. 2016.

[67] Safia Kalwar, Eugenio Gianniti, Joas Yannick Kinouani, Youssef Ridene, and Danilo Ardagna. "Performance Degradation and Cost Impact Evaluation of Privacy Preserving Mechanisms in Big Data Systems." In: *New Frontiers in Quantitative Methods in Informatics*. Ed. by Simonetta Balsamo, Andrea Marin, and Enrico Vicario. Vol. 825. Communications in Computer and Information Science. Springer, 2017, pp. 82–96. ISBN: 978-3-319-91631-6. DOI: 10.1007/978-3-319-91632-3_7.

[86] Marzieh Malekimajd, Danilo Ardagna, Michele Ciavotta, Eugenio Gianniti, Mauro Passacantando, and Alessandro Maria Rizzi. "An Optimization Framework for the Capacity Allocation and Admission Control of MapReduce Jobs in Cloud Systems." In: *The Journal of Supercomputing* (May 2018). ISSN: 1573-0484. DOI: 10.1007/s11227-018-2426-2.

Manuscript Organization

This dissertation is organized as follows. To begin with, Chapter 2 outlines related work and the state of the art. Then Chapter 3 provides details about the performance models developed for this research, thus also addressing research question 1. Chapter 4 describes the problem setting and the formulations proposed for optimization, both at design and run time, which entails research question 2. Further on, Chapters 5 and 6 discuss experimental results that validate both performance models and optimization techniques. Alongside validating research questions 1 and 2, they also show results and case studies about research questions 3 and 4. In the end, Chapter 7 draws the conclusions of this dissertation and hints at possible future work.

CHAPTER 7

Conclusions and Future Work

Throughout this dissertation we presented several techniques to address a number of related problems of interest in the field of DIAs. The main goal pursued in this work was the optimal capacity allocation and management of big data deployments with performance guarantees, both at design and run time, yet a lot of attention was dedicated to other aspects that play an enabling role.

In particular, Chapter 3 focuses on the analysis of various performance models obtained via a series of formalisms, such as QNs, SWNs, and ML. Exploring such a wide range of alternatives was made necessary by the varying requirements of the different optimization procedures, specifically in terms of execution times of the searches themselves, as well as accuracy. To begin with, our optimization procedures are initially formulated through mathematical programming. Algebraic models can be easily added as constraints to the formulation, thus enabling the use of KKT conditions, which provide in closed form a sensible starting point for the search. In this case ML methods are perfectly fit for the requirement, as their output is an algebraic formula involving application parameters and system configurations. However, their big disadvantage is a lack of reliability when applied outside of the range covered by the data available at the time of training: since it is not possible to guarantee that the optimal configuration belongs to that domain, even more so at design time, we also adopt simulation-based techniques to complement ML. During the simulation-optimization approach, QNs, SWNs, or dagSim play their role, thanks to both their insensitivity to the evaluation range and accuracy, with relative errors settling within 3 % and 10 % on average, as reported in Chapter 5. Such results positively answer to research question 1.

Alongside their exploitation to support optimization, we also discuss the use of performance models to assess design choices, in line with the topic of research question 4. For example, Section 3.3.2 extends a basic performance model for Hadoop by considering the effects of spot instances failures caused by cloud providers' decisions to reclaim data center capacity under increased load. Chapter 5 also reports the results of some analyses based on this SWN,

with the goal of understanding the possible cost savings and the corresponding performance degradation enabled by this peculiar pricing model.

A further contribution of Chapter 3 is a pair of approaches to performance modeling of CNN applications deployed on GPGPUs. In this scenario, due to the extreme level of parallelism that rules out most simulation-based formalisms, we opt for two alternative ML models. The first one, the so called per layer model, is based on the derivation of the computational complexity of each type of layer used in CNNs, so as to apply linear regression and link complexity to layer execution times. In this way we obtain a set of models general enough to be applied to new CNNs that are not part of the training set. As a second approach, we also motivate the adoption of an end to end model based on two fundamental parameters, batch size and number of iterations, in order to predict performance when an application is already deployed on a specific system. This method enables a higher accuracy, but at the expense of decreased generality, since the obtained model is hardwired to a particular pair CNN-GPU. In both cases, the observed accuracy is good and, along the lines of research question 1, these performance models may be adopted in future work at the base of optimization procedures.

After presenting performance models, in Chapter 4 we shifted to the optimization methods topic, following research question 2. Thanks to the collaboration with two research projects, DICE H2020 and EUBra-BIGSEA, we investigated both design and run time problems. At first, we focused on design time issues, such as capacity allocation in cloud environments, with a design space exploration that considers also the choice among different alternative VM types to support the workload. After performing the basic choice on the instance type via ML models, the optimization proceeds with a search technique relying on third party simulators and terminates by returning the minimum cost configuration able to satisfy QoS constraints. Later on, we turned our attention to run time problems, like minimizing the overall tardiness of a set of jobs when the current workload exceeds what had been forecast, for instance due to an unforeseen peak of requests. In this scenario there are stringent constraints on the execution time of the optimization procedure, hence we need to consider only the most performant simulators to support the search phase. For this reason, in this case we adopt dagSim or Lundstrom, fast simulators developed by research partners in the frame of EUBra-BIGSEA.

We evaluated these optimization methods in Chapter 6. D-SPACE4Cloud allows for several considerations at design time, for instance assessing the effect of QoS constraints or concurrency levels on the costs incurred to support big data applications. Properly deciding the VM type where to run a workload leads to cost savings up to 36.4 %, with the additional consideration that it is not possible to single out one dominant choice but, as hinted in research question 3, actual workloads, concurrency levels, and QoS constraints play a role in determining the optimal configuration. In order to highlight how D-SPACE4Cloud can be useful during DIA design, we reported the outcomes of a case study for NETF Big Blu, a tax fraud detection application, showing how our tool was used to investigate in early stages the possible impact on costs of different privacy preserving mechanisms: this case study is another example of application of the proposed techniques in designing DIAs, according to the focus of research question 4. Shifting focus to run time aspects, we also assessed the quality of the proposed solution for the minimum weighted

tardiness problem. It appears that OPT_JR can sensibly modulate the solution based on system pressure, with the ability to return it within minutes also under a heavy load. Further, such solutions show a 21 % relative error with respect to measurements on a real system.

In the future, the research carried out so far will be extended in various directions. One of these will be the validation of our performance models against ML and DL workloads, an appealing scenario given their widespread adoption in the industry for a range of applications. Furthermore, via extending the underlying performance predictors, also the optimization methods will be made compatible with such new workloads. Another interesting research branch to explore is the optimal sizing of GPU-based systems, possibly not only for CNNs, but also for other applications. Similarly, it will be relevant to extend the investigation beyond one-GPU nodes, taking into account both multi-GPU machines and clusters of such GPU-enabled installations.

Acronyms

AES Advanced Encryption Standard. 119–121

AM analytical model. vi, 6, 22, 25, 27, 31, 50–54, 92, 93, 95, 96, 98, 100

API application programming interface. 4, 13, 20, 21, 72

CDF cumulative distribution function. 87

CNN convolutional neural network. xiii, 2, 4, 5, 7, 22, 25, 31, 32, 41–49, 81, 102–107, 134, 135

CPN colored Petri net. 23

CPU central processing unit. 2, 5, 6, 17, 18, 23, 25–27, 32, 61, 63, 64, 72, 82, 91, 92, 95, 103, 126, 129

CV coefficient of variation. 39, 41

DAG directed acyclic graph. xi, 4, 6, 8, 21, 28, 29, 32, 34, 41, 52, 63, 72, 89–91, 100, 101

DB database. 3, 20, 109, 112, 118, 119, 125

DDSM DICE Platform, Technology, and Deployment Specific Model. xi, 58, 60, 61

DES discrete event simulator. 7, 60, 72, 89

DIA data intensive application. v, 1–10, 13, 22, 34, 58–61, 63, 69, 71, 79, 89, 97, 132–134

DL deep learning. 4–7, 18, 25, 41, 135

DPIM DICE Platform Independent Model. 58

DT decision tree. 27

DTSM DICE Platform and Technology Specific Model. 58, 60

FCR finite capacity region. 33, 34

FIFO first in, first out. 5, 7, 18, 32–34, 36

FSPN fluid stochastic Petri net. xi, 37, 38

GPGPU general purpose graphics processing unit. 5–7, 25, 42, 43, 46–49, 107, 134

GPS Global Positioning System. 123, 124

GPU graphics processing unit. xiii, 18, 25, 26, 28, 43, 46, 48, 81, 103, 104, 107, 134, 135

HDFS Hadoop Distributed File System. xi, 14–17, 21, 23, 82

HQL Hive Query Language. 20, 23

I/O input/output. 3, 18, 24

IaaS infrastructure as a service. 22, 61, 62, 66, 74

ICT information and communication technology. 22

IDE integrated development environment. 58, 59, 79

IT information technology. 2

JSON JavaScript object notation. 20

KB knowledge base. 27, 50, 51, 53, 54, 93, 98

KKT Karush-Kuhn-Tucker. 54, 67, 75, 77, 79, 133

MAPE mean average percentage error. 27, 51, 53, 93, 95, 97, 100, 101, 103–107

MC Markov chain. xi, 23, 24, 32, 37, 40

MINLP mixed integer nonlinear programming. 60, 66

ML machine learning. v, vi, 6–8, 13, 18–20, 25–27, 31, 32, 50–55, 60, 63, 64, 66, 67, 74, 75, 77, 79, 92, 93, 95–102, 106, 107, 114, 123, 133–135

MVP minimum viable product. 117, 132

NN neural network. 4, 26–28

OS operating system. 14, 15, 24

P8 POWER8. 7, 126

PBS Portable Batch System. 82

PDF probability distribution function. 39, 84, 98

PN Petri net. 23, 25

POSIX Portable Operating System Interface. 15

QN queueing network. v, vi, xi, xiii, 7, 8, 23–25, 27, 29, 31–34, 52, 54, 55, 60, 81, 83–85, 98, 99, 106, 133

QoS quality of service. v, vi, 1, 2, 4, 8, 9, 26–28, 58–60, 62, 71–74, 112, 115, 125, 132, 134

RAM random access memory. 82, 104, 109, 119, 124, 126

RDD resilient distributed dataset. 4, 6, 10, 20, 21

RL reinforcement learning. 27

SLA service level agreement. 8, 10, 23, 26, 29, 74, 79, 107

SPN stochastic Petri net. 8, 23, 37, 60

SQL Structured Query Language. 19, 20, 24, 89

SSD solid state drive. 3, 126

SVM support vector machine. 49

SVR support vector regression. 26, 52, 53, 60, 67, 92, 93

SWN stochastic well formed net. v, vi, xi, xiii, 7, 31–36, 54, 55, 81, 83–86, 106, 107, 133

TAS Transactional Auto Scaler. 27

UML Unified Modeling Language. 58, 60

vCPU virtual central processing unit. 78, 81, 109, 119, 126

VM virtual machine. vi, 7, 8, 10, 34–37, 50, 59–70, 72–75, 77–79, 81, 85, 86, 91, 109, 110, 112, 115, 116, 119, 123–126, 130–132, 134

YARN Yet Another Resource Negotiator. 6, 8, 17, 18, 20, 28, 33, 36, 38, 51, 61, 63, 71, 86

Bibliography

[1] Longendri Aguilera-Mendoza and Monica T. Llorente-Quesada. "Modeling and Simulation of Hadoop Distributed File System in a Cluster of Workstations." English. In: *Model and Data Engineering*. Vol. 8216. 2013, pp. 1–12. ISBN: 978-3-642-41365-0.

[2] Syed Thouheed Ahmed and Dmitri Loguinov. "On the Performance of MapReduce: A Stochastic Approach." In: *IEEE International Conference on Big Data*. IEEE. 2014, pp. 49–54.

[3] A. Brito et al. *D3.4 EUBra-BIGSEA QoS Infrastructure Services Intermediate Version*. 2017. URL: http://www.eubra-bigsea.eu/sites/default/files/D3.4%20EUBra-BIGSEA%20QoS%20infrastructure%20services.pdf.

[4] A. Aleti, B. Buhnova, L. Grunske, A. Koziolek, and I. Meedeniya. "Software Architecture Optimization Methods: A Systematic Literature Review." In: *Software Engineering, IEEE Transactions on* PP.99 (2013), pp. 1–1.

[5] Hanieh Alipour, Yan Liu, Abdelwahab Hamou-Lhadj, and Ian Gorton. "Model Driven Performance Simulation of Cloud Provisioned Hadoop MapReduce Applications." In: *Proceedings of the 8th International Workshop on Modeling in Software Engineering*. MiSE '16. Austin, Texas: ACM, 2016, pp. 48–54. ISBN: 978-1-4503-4164-6. DOI: 10.1145/2896982.2896989.

[6] Omid Alipourfard, Hongqiang Harry Liu, Jianshu Chen, Shivaram Venkataraman, Minlan Yu, and Ming Zhang. "CherryPick: Adaptively Unearthing the Best Cloud Configurations for Big Data Analytics." In: *NSDI*. 2017.

[7] Jussara Almeida, Danilo Ardagna, Igor Ataíde, Enrico Barbierato, Ignacio Blanquer, Sergio López, Túlio Braga, Andrey Brito, Ana Paula Couto, Athanasia Evangelinou, Iury Ferreira, Armstrong Goes, Marco Gribaudo, Raffaela Mirandola, and Fábio Morais. *D3.5 EUBra-BIGSEA QoS Infrastructure Services Final Version*. Oct. 2017. URL: http://eubra-bigsea.eu/sites/default/files/D3.5%20EUBra-BIGSEA%20QoS%20infrastructure%20services%20final%20version.pdf.

[8] Danilo Ardagna, Enrico Barbierato, Athanasia Evangelinou, Eugenio Gianniti, Marco Gribaudo, Túlio B. M. Pinto, Anna Guimarães, Ana Paula Couto da Silva, and Jussara M. Almeida. "Performance Prediction of Cloud-Based Big Data Applications." In: *Proceedings of the 2018 ACM/SPEC International Conference on Performance Engineering*. ICPE

'18. Berlin, Germany: ACM, 2018, pp. 192–199. ISBN: 978-1-4503-5095-2. DOI: 10.1145/3184407.3184420.

[9] Danilo Ardagna, Simona Bernardi, Eugenio Gianniti, Soroush Karimian Aliabadi, Diego Perez-Palacin, and José Ignacio Requeno. "Modeling Performance of Hadoop Applications: A Journey from Queueing Networks to Stochastic Well Formed Nets." In: *16th International Conference on Algorithms and Architectures for Parallel Processing*. Ed. by Jesús Carretero, Javier García Blas, Ryan K. L. Ko, Peter Mueller, and Koji Nakano. Vol. 10048. Lecture Notes in Computer Science. Springer, Dec. 2016, pp. 599–613. ISBN: 978-3-319-49582-8. DOI: 10.1007/978-3-319-49583-5_47.

[10] Danilo Ardagna, Michele Ciavotta, and Mauro Passacantando. "Generalized Nash Equilibria for the Service Provisioning Problem in Multi-Cloud Systems." In: *IEEE Trans. Services Computing* 10.3 (2017), pp. 381–395.

[11] Danilo Ardagna, Carlo Ghezzi, and Raffaela Mirandola. "Rethinking the Use of Models in Software Architecture." In: *QoSA*. 2008.

[12] Danilo Ardagna, Eugenio Gianniti, Jacopo Rigoli, Safia Kalwar, and Giuliano Casale. *DICE Optimization Tools — Final Version*. July 2017. URL: http://wp.doc.ic.ac.uk/dice-h2020/wp-content/uploads/sites/75/2017/08/D3.9_DICE-optimization-tools-Final-version.pdf.

[13] Sylvain Arlot and Alain Celisse. "A Survey of Cross-Validation Procedures for Model Selection." In: *Statistics Surveys*. 2010, pp. 40–79.

[14] Matej Artac, Tadej Borovsak, Elisabetta Di Nitto, Michele Guerriero, and Damian Andrew Tamburri. "Model-Driven Continuous Deployment for Quality DevOps." In: *QUDOS*. 2016.

[15] Ehsan Ataie, Eugenio Gianniti, Danilo Ardagna, and Ali Movaghar. "A Combined Analytical Modeling Machine Learning Approach for Performance Prediction of MapReduce Jobs in Cloud Environment." In: *SYNASC* (Timisoara, Romania). 2016.

[16] Soheib Baarir, Marco Beccuti, Davide Cerotti, Massimiliano De Pierro, Susanna Donatelli, and Giuliana Franceschinis. "The GreatSPN Tool: Recent Enhancements." In: *ACM SIGMETRICS PER* 36.4 (2009), pp. 4–9.

[17] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate." In: *CoRR* abs/1409.0473 (2014).

[18] Soheil Bahrampour, Naveen Ramakrishnan, Lukas Schott, and Mohak Shah. "Comparative Study of Caffe, Neon, Theano, and Torch for Deep Learning." In: *CoRR* abs/1511.06435 (2015).

[19] Simonetta Balsamo, Peter G. Harrison, and Andrea Marin. "Methodological Construction of Product-Form Stochastic Petri Nets for Performance Evaluation." In: *Journal of Systems and Software* 85.7 (2012), pp. 1520–1539. DOI: 10.1016/j.jss.2011.11.1042.

[20] Enrico Barbierato, Marco Gribaudo, and Mauro Iacono. "Modeling Apache Hive Based Applications in Big Data Architectures." In: *Proceedings of the 7th International Conference on Performance Evaluation Methodologies and Tools*. ValueTools '13. Torino, Italy: ICST, 2013, pp. 30–38. ISBN: 978-1-936968-48-0. DOI: 10.4108/icst.valuetools.2013.254398.

[21] Enrico Barbierato, Marco Gribaudo, and Mauro Iacono. "Modeling Hybrid Systems in SIMTHESys." In: *Proceedings of the 8th International Workshop on Practical Applications of Stochastic Modelling*. Münster, Germany, 2016.

[22] Shouvik Bardhan and Daniel A. Menascé. "Queuing Network Models to Predict the Completion Time of the Map Phase of MapReduce Jobs." In: *Proc. International Computer Measurement Group Conf.* (Las Vegas, NV). Dec. 2012.

[23] S. Becker, H. Koziolek, and R. Reussner. "The Palladio Component Model for Model-driven Performance Prediction." In: *Journal of Systems and Software* 82.1 (2009), pp. 3–22.

[24] Marco Bertoli, Giuliano Casale, and Giuseppe Serazzi. "JMT: Performance Engineering Tools for System Modeling." In: *SIGMETRICS Perform. Eval. Rev.* 36.4 (2009), pp. 10–15. ISSN: 0163-5999. DOI: 10.1145/1530873.1530877.

[25] F. Brosig, P. Meier, S. Becker, A. Koziolek, H. Koziolek, and S. Kounev. "Quantitative Evaluation of Model-Driven Performance Analysis and Simulation of Component-Based Architectures." In: *Software Engineering, IEEE Transactions on* 41.2 (Feb. 2015), pp. 157–175. ISSN: 0098-5589. DOI: 10.1109/TSE.2014.2362755.

[26] Dario Bruneo, Francesco Longo, Rahul Ghosh, Marco Scarpa, Antonio Puliafito, and Kishor S. Trivedi. "Analytical Modeling of Reactive Autonomic Management Techniques in IaaS Clouds." In: *IEEE CLOUD*. 2015.

[27] Giuliano Casale, Danilo Ardagna, Matej Artac, Franck Barbier, Elisabetta Di Nitto, Alexis Henry, Gabriel Iuhasz, Christophe Joubert, José Merseguer, Victor Ion Munteanu, Juan Fernando Pérez, Dana Petcu, Matteo Rossi, Craig Sheridan, Ilias Spais, and Daniel Vladuic. "DICE: Quality-Driven Development of Data-Intensive Cloud Applications." In: *International Workshop on Modeling in Software Engineering* (Florence, Italy). Ed. by Jeff Gray, Marsha Chechik, Vinay Kulkarni, and Richard F. Paige. IEEE Computer Society, May 2015, pp. 78–83. ISBN: 978-1-4673-7055-4. DOI: 10.1109/MiSE.2015.21. URL: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7166141.

[28] Aniello Castiglione, Marco Gribaudo, Mauro Iacono, and Francesco Palmieri. "Exploiting Mean Field Analysis to Model Performances of Big Data Architectures." In: *Future Generation Computer Systems* 37 (2014), pp. 203–211. ISSN: 0167-739X. DOI: 10.1016/j.future.2013.07.016.

[29] C.-C. Chang and C.-J. Lin. "LIBSVM: A Library for Support Vector Machines." In: *ACM Transactions on Intelligent Systems and Technology* 2.27 (2011), pp. 1–27.

[30] Tatsuhiro Chiba and Tamiya Onodera. "Workload Characterization and Optimization of TPC-H Queries on Apache Spark." In: *ISPASS*. IEEE, Apr. 2016. DOI: 10.1109/ISPASS.2016.7482079.

[31] Wesley W. Chu, Chi-Man Sit, and Kin K. Leung. "Task Response Time for Real-Time Distributed Systems with Resource Contentions." In: *IEEE Trans. Softw. Eng.* 17.10 (), pp. 1076–1092. ISSN: 0098-5589. DOI: 10.1109/32.99195.

[32] Michele Ciavotta, Eugenio Gianniti, and Danilo Ardagna. "Capacity Allocation for Big Data Applications in the Cloud." In: *ICPE'17 Companion* (L'Aquila, Italy). Apr. 2017, pp. 175–176.

[33] Michele Ciavotta, Eugenio Gianniti, and Danilo Ardagna. "D-SPACE4Cloud: A Design Tool for Big Data Applications." In: *ICA3PP* (Granada, Spain). Lecture Notes in Computer Science 10048. Springer, Dec. 2016, pp. 614–629.

[34] Valentin Dalibard, Michael Schaarschmidt, and Eiko Yoneki. "BOAT: Building Auto-Tuners with Structured Bayesian Optimization." In: *WWW*. 2017.

[35] J. Dean and S. Ghemawat. "MapReduce: Simplified Data Processing on Large Clusters." In: *Commun. ACM* 51.1 (2008), pp. 107–113.

[36] Jeffrey Dean and Sanjay Ghemawat. "MapReduce: Simplified Data Processing on Large Clusters." In: *6th Symposium on Operating Systems Design and Implementation*. 2004, pp. 137–149.

[37] Christina Delimitrou and Christos Kozyrakis. "Paragon: QoS-aware Scheduling for Heterogeneous Datacenters." In: *SIGPLAN Not.* 48.4 (Mar. 2013), pp. 77–88. ISSN: 0362-1340.

[38] Christina Delimitrou and Christos Kozyrakis. "Quasar: Resource-Efficient and QoS-Aware Cluster Management." In: *ACM SIGPLAN Notices*. Vol. 49. 4. ACM. 2014, pp. 127–144.

[39] Christina Delimitrou and Christos Kozyrakis. "Quasar: Resource-efficient and QoS-aware Cluster Management." In: *ASPLOS*. 2014.

[40] H. Derrick. *Survey Shows Huge Popularity Spike for Apache Spark*. 2015. URL: http://fortune.com/2015/09/25/apache-spark-survey.

[41] P Di Sanzo, F Quaglia, B Ciciani, A Pellegrini, D Didona, P Romano, R Palmieri, and S Peluso. "A Flexible Framework for Accurate Simulation of Cloud In-Memory Data Stores." In: *Simulation Modelling Practice and Theory* 58 (2015), pp. 219–238.

[42] Diego Didona, Pascal Felber, Derin Harmanci, Paolo Romano, and Jörg Schenker. "Identifying the Optimal Level of Parallelism in Transactional Memory Applications." In: *Computing* 97.9 (2015), pp. 939–959.

[43] Diego Didona, Francesco Quaglia, Paolo Romano, and Ennio Torre. "Enhancing Performance Prediction Robustness by Combining Analytical Modeling and Machine Learning." In: *6th ACM/SPEC International Conference on Performance Engineering*. 2015, pp. 145–156.

[44] Diego Didona and Paolo Romano. "Hybrid Machine Learning/Analytical Models for Performance Prediction: A Tutorial." In: *International Conference on Performance Engineering*. ACM/SPEC. 2015, pp. 341–344.

[45] Diego Didona and Paolo Romano. "On Bootstrapping Machine Learning Performance Predictors via Analytical Models." In: (2014). arXiv: 1410.5102v1 [cs.PF].

[46] Diego Didona, Paolo Romano, Sebastiano Peluso, and Francesco Quaglia. "Transactional Auto Scaler: Elastic Scaling of Replicated In-Memory Transactional Data Grids." In: *ACM Transactions on Autonomous and Adaptive Systems* 9.2 (2014), pp. 1–32.

[47] Daniel J. Dubois and Giuliano Casale. "OptiSpot: Minimizing Application Deployment Cost Using Spot Cloud Resources." In: *Cluster Computing* (2016), pp. 1–17.

[48] John Ganz and David Reinsel. *The Digital Universe in 2020: Big Data, Bigger Digital Shadows, and Biggest Growth in the Far East — United States*. Tech. rep. IDC, Feb. 2013. URL: https://www.emc.com/collateral/analyst-reports/idc-digital-universe-united-states.pdf (visited on 04/07/2015).

[49] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1990. ISBN: 0716710455.

[50] Eugenio Gianniti, Danilo Ardagna, Michele Ciavotta, and Mauro Passacantando. "A Game-Theoretic Approach for Runtime Capacity Allocation in MapReduce." In: *CCGrid* (Madrid, Spain). May 2017, pp. 1080–1089.

[51] Eugenio Gianniti, Alessandro Maria Rizzi, Enrico Barbierato, Marco Gribaudo, and Danilo Ardagna. "Fluid Petri Nets for the Performance Evaluation of MapReduce and Spark Applications." In: *ACM Performance Evaluation Review* 44.4 (Mar. 2017), pp. 23–36. DOI: 10.1145/3092819.3092824.

[52] Eugenio Gianniti, Alessandro Maria Rizzi, Enrico Barbierato, Marco Gribaudo, and Danilo Ardagna. "Fluid Petri Nets for the Performance Evaluation of MapReduce Applications." In: *InfQ* (Taormina, Italy). Oct. 2016.

[53] G. P. Gibilisco, M. Li, L. Zhang, and D. Ardagna. "Stage Aware Performance Modeling of DAG Based in Memory Analytic Platforms." In: *Cloud*. 2016.

[54] M. A. Greene and K. Sreekanti. *Big Data in the Enterprise: We Need an "Easy Button" for Hadoop*. 2016. URL: http://www.oreilly.com/pub/e/3643.

[55] Marco Gribaudo and Miklós Telek. "Fluid Models in Performance Analysis." In: *International School on Formal Methods for the Design of Computer, Communication, and Software Systems*. Ed. by Marco Bernardo and Jane Hillston. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 271–317. ISBN: 978-3-540-72522-0. DOI: 10.1007/978-3-540-72522-0_7.

[56] Lei Gu and Huan Li. "Memory or Time: Performance Evaluation for Iterative Operation on Hadoop and Spark." In: *HPCC-EUC*. IEEE, Nov. 2013. DOI: 10.1109/HPCC.and.EUC.2013.106.

[57] Stefan Hadjis, Ce Zhang, Ioannis Mitliagkas, and Christopher Ré. "Omnivore: An Optimizer for Multi-Device Deep Learning on CPUs and GPUs." In: *CoRR* abs/1606.04487 (2016).

[58] Herodotos Herodotou, Fei Dong, and Shivnath Babu. "No One (Cluster) Size Fits All: Automatic Cluster Sizing for Data-intensive Analytics." In: *SOCC*. Cascais, Portugal, 2011. DOI: 10.1145/2038916.2038934.

[59] Herodotos Herodotou, Harold Lim, Gang Luo, Nedyalko Borisov, Liang Dong, Fatma Bilgen Cetin, and Shivnath Babu. "Starfish: A Self-Tuning System for Big Data Analytics." In: *CIDR*. 2011.

[60] Benjamin Hindman, Andy Konwinski, Matei Zaharia, Ali Ghodsi, Anthony D. Joseph, Randy Katz, Scott Shenker, and Ion Stoica. "Mesos: A Platform for Fine-Grained Resource Sharing in the Data Center." In: *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*. NSDI'11. Boston, MA: USENIX Association, 2011, pp. 295–308. URL: http://dl.acm.org/citation.cfm?id=1972457.1972488.

[61] Engin Ipek, Sally A. McKee, Bronis R. de Supinski, and Rich Caruana. "Efficiently Exploring Architectural Design Spaces via Predictive Modeling." In: *12th International Conference on Architectural Support for Programming Languages and Operating Systems*. 2006, pp. 195–206.

[62] Florin Isaila, Prasanna Balaprakash, Stefan M Wild, Dries Kimpe, Rob Latham, Rob Ross, and Paul Hovland. "Collective I/O Tuning Using Analytical and Machine Learning Models." In: *International Conference on Cluster Computing*. IEEE. 2015, pp. 128–137.

[63] H. V. Jagadish, Johannes Gehrke, Alexandros Labrinidis, Yannis Papakonstantinou, Jignesh M. Patel, Raghu Ramakrishnan, and Cyrus Shahabi. "Big Data and Its Technical Challenges." In: *Commun. ACM* 57.7 (July 2014), pp. 86–94. ISSN: 0001-0782.

[64] Kurt Jensen, Lars Michael Kristensen, and Lisa Wells. "Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems." In: *International Journal on Software Tools for Technology Transfer* 9.3-4 (2007), pp. 213–254.

[65] Wenhao Jia, Kelly A. Shaw, and Margaret Martonosi. "Stargazer: Automated Regression-based GPU Design Space Exploration." In: *ISPASS*. IEEE. Apr. 2012. DOI: 10.1109/ISPASS.2012.6189201.

[66] Hui Jin, Kan Qiao, Xian-He Sun, and Ying Li. "Performance under Failures of MapReduce Applications." In: *CCGrid*. 2011.

[67] Safia Kalwar, Eugenio Gianniti, Joas Yannick Kinouani, Youssef Ridene, and Danilo Ardagna. "Performance Degradation and Cost Impact Evaluation of Privacy Preserving Mechanisms in Big Data Systems." In: *New Frontiers in Quantitative Methods in Informatics*. Ed. by Simonetta Balsamo, Andrea Marin, and Enrico Vicario. Vol. 825. Communications in Computer and Information Science. Springer, 2017, pp. 82–96. ISBN: 978-3-319-91631-6. DOI: 10.1007/978-3-319-91632-3_7.

[68] Karthik Kambatla, Giorgos Kollias, Vipin Kumar, and Ananth Grama. "Trends in Big Data Analytics." In: *Journal of Parallel and Distributed Computing* 74.7 (2014), pp. 2561–2573. ISSN: 0743-7315. DOI: 10.1016/j.jpdc.2014.01.003.

[69] Andrew Kerr, Gregory Diamos, and Sudhakar Yalamanchili. "Modeling GPU-CPU Workloads and Systems." In: *Proceedings of the 3rd Workshop on General-Purpose Computation on Graphics Processing Units*. GPGPU-3. Pittsburgh, Pennsylvania, USA: ACM, 2010, pp. 31–42. ISBN: 978-1-60558-935-0. DOI: 10.1145/1735688.1735696.

[70] Anne Koziolek, Heiko Koziolek, and Ralf Reussner. "PerOpteryx: Automated Application of Tactics in Multi-objective Software Architecture Optimization." In: *QoSA*. QoSA-ISARCS '11. Boulder, Colorado, USA: ACM, 2011, pp. 33–42. ISBN: 978-1-4503-0724-6. DOI: 10.1145/2000259.2000267.

[71] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks." In: *NIPS*. Vol. 1. Dec. 2012, pp. 1097–1105.

[72] Johannes Kross and Helmut Krcmar. "Model-Based Performance Evaluation of Batch and Stream Applications for Big Data." In: *MASCOTS*. 2017.

[73] Palden Lama and Xiaobo Zhou. "AROMA: Automated Resource Allocation and Configuration of MapReduce Environment in the Cloud." In: *9th International Conference on Autonomic Computing*. 2012, pp. 63–72.

[74] Douglas Laney. *3D Data Management: Controlling Data Volume, Velocity, and Variety*. Tech. rep. META Group, 2012.

[75] Edward D. Lazowska, John Zahorjan, G. Scott Graham, and Kenneth C. Sevcik. *Quantitative System Performance. Computer System Analysis Using Queueing Network Models*. Prentice-Hall, 1984. URL: http://homes.cs.washington.edu/~lazowska/qsp/ (visited on 04/07/2015).

[76] Kyong-Ha Lee, Yoon-Joon Lee, Hyunsik Choi, Yon Dohn Chung, and Bongki Moon. "Parallel Data Processing with MapReduce: A Survey." In: *SIGMOD Rec.* 40.4 (Jan. 2012), pp. 11–20. ISSN: 0163-5808. DOI: 10.1145/2094114.2094118.

[77] Min Li, Jian Tan, Yandong Wang, Li Zhang, and Valentina Salapura. "SparkBench: A Comprehensive Benchmarking Suite for in Memory Data Analytic Platform Spark." In: *Proceedings of the 12th ACM International Conference on Computing Frontiers*. CF '15. Ischia, Italy: ACM, 2015, 53:1–53:8. ISBN: 978-1-4503-3358-0. DOI: 10.1145/2742854.2747283.

[78] De-Ron Liang and Satish K. Tripathi. "On Performance Prediction of Parallel Computations with Precedent Constraints." In: *IEEE Transactions on Parallel and Distributed Systems* 11.5 (May 2000), pp. 491–508. ISSN: 1045-9219. DOI: 10.1109/71.852402.

[79] Min Lin, Qiang Chen, and Shuicheng Yan. "Network in Network." In: *CoRR* abs/1312.4400 (2013).

[80] Minghong Lin, Li Zhang, Adam Wierman, and Jian Tan. "Joint Optimization of Overlapping Phases in MapReduce." In: *SIGMETRICS Performance Evaluation Review* 41.3 (2013), pp. 16–18. DOI: 10.1145/2567529.2567534.

[81] Xuelian Lin, Zide Meng, Chuan Xu, and Meng Wang. "A Practical Performance Model for Hadoop MapReduce." In: *CLUSTER*. IEEE. 2012, pp. 231–239.

[82] Xuelian Lin, Zide Meng, Chuan Xu, and Meng Wang. "A Practical Performance Model for Hadoop MapReduce." In: *International Conference on Cluster Computing Workshops*. IEEE. 2012. DOI: 10.1109/ClusterW.2012.24.

[83] Weiguo Liu, Wolfgang Muller-Wittig, and Bertil Schmidt. "Performance Predictions for General-Purpose Computation on GPUs." In: *ICPP*. IEEE. Sept. 2007. DOI: 10.1109/ICPP.2007.67.

[84] Chi-Keung Luk, Sunpyo Hong, and Hyesoon Kim. "Qilin: Exploiting Parallelism on Heterogeneous Multiprocessors with Adaptive Mapping." In: *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*. MICRO 42. New York, New York: ACM, 2009, pp. 45–55. ISBN: 978-1-60558-798-1. DOI: 10.1145/1669112.1669121.

[85] V. W. Mak and S. F. Lundstrom. "Predicting Performance of Parallel Computations." In: *IEEE Trans. Parallel Distrib. Syst.* 1.3 (July 1990), pp. 257–270. ISSN: 1045-9219. DOI: 10.1109/71.80155.

[86] Marzieh Malekimajd, Danilo Ardagna, Michele Ciavotta, Eugenio Gianniti, Mauro Passacantando, and Alessandro Maria Rizzi. "An Optimization Framework for the Capacity Allocation and Admission Control of MapReduce Jobs in Cloud Systems." In: *The Journal of Supercomputing* (May 2018). ISSN: 1573-0484. DOI: 10.1007/s11227-018-2426-2.

[87] Marzieh Malekimajd, Danilo Ardagna, Michele Ciavotta, Alessandro Maria Rizzi, and Mauro Passacantando. "Optimal Map Reduce Job Capacity Allocation in Cloud Systems." In: *SIGMETRICS Perform. Eval. Rev.* 42.4 (June 2015), pp. 51–61. ISSN: 0163-5999. DOI: 10.1145/2788402.2788410.

[88] João Eugenio Marynowski, Altair Olivo Santin, and Andrey Ricardo Pimentel. "Method for Testing the Fault Tolerance of MapReduce Frameworks." In: *Computer Networks* 86 (2015), pp. 1–13.

[89] Ilias Mavridis and Helen Karatza. "Performance Evaluation of Cloud-Based Log File Analysis with Apache Hadoop and Apache Spark." In: *Journal of Systems and Software* 125 (2017), pp. 133–151. ISSN: 0164-1212. DOI: 10.1016/j.jss.2016.11.037.

[90] Rizwan Mian, Patrick Martin, and Jose Luis Vazquez-Poletti. "Provisioning Data Analytic Workloads in a Cloud." In: *Future Gener. Comput. Syst.* 29.6 (Aug. 2013), pp. 1452–1458. ISSN: 0167-739X.

[91] K. Morton, A Friesen, M. Balazinska, and D. Grossman. "Estimating the Progress of MapReduce Pipelines." In: *ICDE*. 2010.

[92] Kristi Morton, Magdalena Balazinska, and Dan Grossman. "ParaTimer: A Progress Indicator for MapReduce DAGs." In: *SIGMOD*. 2010. DOI: 10.1145/1807167.1807223.

[93] Randolf D. Nelson and Asser N. Tantawi. "Approximate Analysis of Fork/Join Synchronization in Parallel Queues." In: *IEEE Trans. Computers* 37.6 (1988), pp. 739–743. DOI: 10.1109/12.2213.

[94] OMG. *PEPA: Performance Evaluation Process Algebra*. 2015. URL: http://www.dcs.ed.ac.uk/pepa/tools/.

[95] Rasha Osman and Peter G. Harrison. "Approximating Closed Fork-Join Queueing Networks Using Product-Form Stochastic Petri-Nets." In: *J. Syst. Softw.* 110.C (Dec. 2015), pp. 264–278. ISSN: 0164-1212. DOI: 10.1016/j.jss.2015.08.036.

[96] Kay Ousterhout, Ryan Rasti, Sylvia Ratnasamy, Scott Shenker, and Byung Gon Chun. "Making Sense of Performance in Data Analytics Frameworks." In: *NSDI* (2015).

[97] Xinghao Pan, Shivaram Venkataraman, Zizheng Tai, and Joseph Gonzalez. "Hemingway: Modeling Distributed Optimization Algorithms." In: *CoRR* abs/1702.05865 (2017).

[98] Linh T. X. Phan, Zhuoyao Zhang, Qi Zheng, Boon Thau Loo, and Insup Lee. "An Empirical Analysis of Scheduling Techniques for Real-Time Cloud-based Data Processing." In: *SOCA*. 2011. ISBN: 978-1-4673-0318-7. DOI: 10.1109/SOCA.2011.6166240.

[99] Michael L. Pinedo. *Scheduling: Theory, Algorithms, and Systems*. 3rd. Springer Publishing Company, Incorporated, 2008. ISBN: 978-1-4614-2361-4.

[100] J. Polo, D. Carrera, Y. Becerra, J. Torres, E. Ayguadé, M. Steinder, and I Whalley. "Performance-driven Task Co-scheduling for MapReduce Environments." In: *NOMS*. 2010.

[101] Jorda Polo, Yolanda Becerra, David Carrera, Malgorzata Steinder, Ian Whalley, Jordi Torres, and Eduard Ayguadé. "Deadline-Based MapReduce Workload Management." In: *IEEE Trans. Network and Service Management* 10.2 (2013), pp. 231–244. DOI: 10.1109/TNSM.2012.122112.110163.

[102] B. Thirumala Rao and L. S. S. Reddy. "Survey on Improved Scheduling in Hadoop MapReduce in Cloud Environments." In: *CoRR* abs/1207.0780 (2012).

[103] Alessandro Maria Rizzi. "Support Vector Regression Model for Big-Data Systems." In: (Dec. 2016). arXiv: 1612.01458 [cs.DC].

[104] Diego Rughetti, Pierangelo Di Sanzo, Bruno Ciciani, Francesco Quaglia, and Sapienza Universit. "Analytical/ML Mixed Approach for Concurrency Regulation in Software Transactional Memory." In: *Cluster, Cloud and Grid Computing*. CCGrid. 2014, pp. 81–91.

[105] M Carmen Ruiz, Javier Calleja, and Diego Cazorla. "Petri Nets Formalization of Map/Reduce Paradigm to Optimise the Performance-Cost Tradeoff." In: *Trustcom/BigDataSE/ISPA*. Vol. 3. IEEE. 2015, pp. 92–99.

[106] Bikas Saha, Hitesh Shah, Siddharth Seth, Gopal Vijayaraghavan, Arun Murthy, and Carlo Curino. "Apache Tez: A Unifying Framework for Modeling and Building Data Processing Applications." In: *International Conference on Management of Data*. SIGMOD '15. ACM. Melbourne, Victoria, Australia, 2015, pp. 1357–1369. ISBN: 978-1-4503-2758-9. DOI: 10.1145/2723372.2742790.

[107] Tara N. Sainath, Brian Kingsbury, George Saon, Hagen Soltau, Abdelrahman Mohamed, George E. Dahl, and Bhuvana Ramabhadran. "Deep Convolutional Neural Networks for Large-scale Speech Tasks." In: *Neural Networks* 64 (2015), pp. 39–48.

[108] Malte Schwarzkopf, Andy Konwinski, Michael Abd-El-Malek, and John Wilkes. "Omega: Flexible, Scalable Schedulers for Large Compute Clusters." In: *SIGOPS European Conference on Computer Systems*. EuroSys. Prague, Czech Republic, 2013, pp. 351–364. URL: http://eurosys2013.tudos.org/wp-content/uploads/2013/paper/Schwarzkopf.pdf.

[109] Carter Shanklin. *Benchmarking Apache Hive 13 for Enterprise Hadoop*. June 2, 2014. URL: https://hortonworks.com/blog/benchmarking-apache-hive-13-enterprise-hadoop/.

[110] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition." In: *ICLR* (San Diego, CA, USA). May 2015. arXiv: 1409.1556v6 [cs.CV].

[111] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting." In: *Journal of Machine Learning Research* 15.1 (2014), pp. 1929–1958.

[112] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going Deeper with Convolutions." In: *CVPR* (Boston, MA, USA). IEEE, June 2015. DOI: 10.1109/CVPR.2015.7298594.

[113] Gerald Tesauro, Nicholas K Jong, Rajarshi Das, and Mohamed N Bennani. "On the Use of Hybrid Reinforcement Learning for Autonomic Resource Allocation." In: *Cluster Computing* 10 (2007), pp. 287–299.

[114] Eno Thereska and Gregory R Ganger. "IRONModel: Robust Performance Models in the Wild." In: *ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*. 2008, pp. 253–264.

[115] Fengguang Tian and Keke Chen. "Towards Optimal Resource Provisioning for Running MapReduce Programs in Public Clouds." In: *CLOUD*. 2011.

[116] Mirco Tribastone, Stephen Gilmore, and Jane Hillston. "Scalable Differential Analysis of Process Algebra Models." In: *IEEE Transactions on Software Engineering* 38.1 (2012), pp. 205–219. ISSN: 0098-5589. DOI: 10.1109/TSE.2010.82.

[117] Vinod Kumar Vavilapalli, Arun C. Murthy, Chris Douglas, Sharad Agarwal, Mahadev Konar, Robert Evans, Thomas Graves, Jason Lowe, Hitesh Shah, Siddharth Seth, Bikas Saha, Carlo Curino, Owen O'Malley, Sanjay Radia, Benjamin Reed, and Eric Baldeschwieler. "Apache Hadoop YARN: Yet Another Resource Negotiator." In: *SOCC*. 2013.

[118] Shivaram Venkataraman, Zongheng Yang, Michael J Franklin, Benjamin Recht, and Ion Stoica. "Ernest: Efficient Performance Prediction for Large-Scale Advanced Analytics." In: *NSDI*. 2016.

[119] Abhishek Verma, Ludmila Cherkasova, and Roy H. Campbell. "ARIA: Automatic Resource Inference and Allocation for MapReduce Environments." In: *Proceedings of the Eighth International Conference on Autonomic Computing*. June 2011.

[120] Abhishek Verma, Ludmila Cherkasova, and Roy H. Campbell. "Profiling and Evaluating Hardware Choices for MapReduce Environments: An Application-aware Approach." In: *Performance Evaluation* 79 (2014), pp. 328–344. DOI: 10.1016/j.peva.2014.07.020.

[121] Abhishek Verma, Luis Pedrosa, Madhukar Korupolu, David Oppenheimer, Eric Tune, and John Wilkes. "Large-Scale Cluster Management at Google with Borg." In: *EuroSys*. 2015.

[122] Emanuel Vianna, Giovanni Comarela, Tatiana Pontes, Jussara M. Almeida, Virgílio A. F. Almeida, Kevin Wilkinson, Harumi A. Kuno, and Umeshwar Dayal. "Analytical Performance Models for MapReduce Workloads." In: *International Journal of Parallel Programming* 41.4 (2013), pp. 495–525. DOI: 10.1007/s10766-012-0227-4.

[123] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. "Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion." In: *Journal of Machine Learning Research* 11 (2010), pp. 3371–3408.

[124] Kewen Wang and Mohammad Maifi Hasan Khan. "Performance Prediction for Apache Spark Platform." In: *HPCC-CSS-ICESS*. IEEE, Aug. 2015. DOI: 10.1109/HPCC-CSS-ICESS.2015.246.

[125] Kewen Wang, Mohammad Maifi Hasan Khan, Nhan Nguyen, and Swapna Gokhale. "Modeling Interference for Apache Spark Jobs." In: *CLOUD*. IEEE, 2016. DOI: 10.1109/CLOUD.2016.0063.

[126] Weina Wang, Kai Zhu, Lei Ying, Jian Tan, and Li Zhang. "MapTask Scheduling in MapReduce With Data Locality: Throughput and Heavy-Traffic Optimality." In: *IEEE/ACM Transactions on Networking* 24.1 (2016), pp. 190–203.

[127] Yandong Wang, Li Zhang, Yufei Ren, and Wei Zhang. "Nexus: Bringing Efficient and Scalable Training to Deep Learning Frameworks." In: *MASCOTS*. Banff, Canada, Sept. 2017, pp. 12–21.

[128] *Worldwide Semiannual Big Data and Analytics Spending Guide*. Mar. 2017. URL: https://www.idc.com/getdoc.jsp?containerId=IDC_P33195.

[129] Xiaoyong Xu and Maolin Tang. "A New Approach to the Cloud-based Heterogeneous MapReduce Placement Problem." In: *IEEE Transactions on Services Computing* 9.6 (2015), pp. 862–871. DOI: 10.1109/TSC.2015.2433914.

[130] Feng Yan, Ludmila Cherkasova, Zhuoyao Zhang, and Evgenia Smirni. "Optimizing Power and Performance Trade-offs of MapReduce Job Processing with Heterogeneous Multi-Core Processors." In: *CLOUD*. 2014. DOI: 10.1109/CLOUD.2014.41.

[131] Xiao Yang and Jianling Sun. "An Analytical Performance Model of MapReduce." In: *CCIS*. IEEE. 2011.

[132] Nezih Yigitbasi, Theodore L. Willke, Guangdeng Liao, and Dick Epema. "Towards Machine Learning-Based Auto-tuning of MapReduce." In: *IEEE 21st International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems*. 2013, pp. 11–20.

[133] Xiaolong Yu and Wei Li. "Performance Modelling and Analysis of MapReduce/Hadoop Workloads." In: *LANMAN*. 2015.

[134] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael J. Franklin, Scott Shenker, and Ion Stoica. "Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing." In: *9th USENIX Conference on Networked Systems Design and Implementation*. NSDI'12. San Jose, CA: USENIX Association, 2012, pp. 2–2. URL: http://dl.acm.org/citation.cfm?id=2228298.2228301.

[135] Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. "Spark: Cluster Computing with Working Sets." In: *2nd USENIX Conference on Hot Topics in Cloud Computing*. HotCloud'10. Boston, MA: USENIX Association, 2010, pp. 10–10. URL: http://dl.acm.org/citation.cfm?id=1863103.1863113.

[136] Matei Zaharia, Patrick Wendell, Andy Konwinski, and Holden Karau. *Learning Spark*. O'Reilly Media, Inc., 2015. ISBN: 978-1-4493-5862-4.

[137] Wei Zhang, Sundaresan Rajasekaran, Shaohua Duan, Timothy Wood, and Mingfa Zhu. "Minimizing Interference and Maximizing Progress for Hadoop Virtual Machines." In: *SIGMETRICS Performance Evaluation Review* 42.4 (2015), pp. 62–71. DOI: 10.1145/2788402.2788411.

[138] Zhuoyao Zhang, Ludmila Cherkasova, and Boon Thau Loo. "Exploiting Cloud Heterogeneity to Optimize Performance and Cost of MapReduce Processing." In: *SIGMETRICS Performance Evaluation Review* 42.4 (2015), pp. 38–50. DOI: 10.1145/2788402.2788409.

[139] Zhuoyao Zhang, Ludmila Cherkasova, and Boon Thau Loo. "Exploiting Cloud Heterogeneity to Optimize Performance and Cost of MapReduce Processing." In: *SIGMETRICS Performance Evaluation Review* 42.4 (2015), pp. 38–50. DOI: 10.1145/2788402.2788409.

[140] Zhuoyao Zhang, Ludmila Cherkasova, Abhishek Verma, and Boon Thau Loo. "Automated Profiling and Resource Management of Pig Programs for Meeting Service Level Objectives." In: *ICAC*. San Jose, California, USA, 2012. ISBN: 978-1-4503-1520-3. DOI: 10.1145/2371536.2371546.