Artificial Intelligence (AI)-Centric Management of Resources in Modern Distributed Computing Systems

Shashikant Ilager[1], Rajeev Muralidhar[1,2], Rajkumar Buyya[1]
[1]*Cloud Computing and Distributed Systems (CLOUDS) Laboratory*
School of Computing and Information Systems
The University of Melbourne, Australia
[2]*Amazon Web Services (AWS), Australia*

Abstract—Contemporary Distributed Computing Systems (DCS) such as Cloud Data Centers are large scale, complex, heterogeneous, and distributed across multiple networks and geographical boundaries. On the other hand, the Internet of Things (IoT)-driven applications are producing a huge amount of data that requires real-time processing and fast response. Managing these resources efficiently to provide reliable services to end-users or applications is a challenging task. The existing Resource Management Systems (RMS) rely on either static or heuristic solutions inadequate for such composite and dynamic systems. The advent of Artificial Intelligence (AI) due to data availability and processing capabilities manifested into possibilities of exploring data-driven solutions in RMS tasks that are adaptive, accurate, and efficient. In this regard, this paper aims to draw the motivations and necessities for data-driven solutions in resource management. It identifies the challenges associated with it and outlines the potential future research directions detailing where and how to apply the data-driven techniques in the different RMS tasks. Finally, it provides a conceptual data-driven RMS model for DCS and presents the two real-time use cases (GPU frequency scaling and data centre resource management from Google Cloud and Microsoft Azure) demonstrating AI-centric approaches' feasibility.

Index Terms—Distributed Computing, Resource Management, AI Techniques, Edge Computing, Cloud Computing

I. INTRODUCTION

Internet-based Distributed Computing Systems (DCS) have become an essential backbone of the modern digital economy, society, and industrial operations. The emergence of the Internet of Things (IoT), diverse mobile applications, smart grids, smart industries, and smart cities has resulted in massive amounts of data generation. Thus, it has increased the demand for computing resources [1] to process this data and derive valuable insights for users and businesses. According to the report from Norton [2], 21 billion IoT devices will be connected to the internet by 2025, creating substantial economic opportunities. Computing models such as Cloud and Edge computing have revolutionised the way services are delivered and consumed by providing flexible on-demand access to services with a pay-as-you-go model. Besides, new application and execution models like micro-services and serverless or Function as Service (FaaS) computing [3] are becoming mainstream that significantly reduces the complexities in the design and deployment of software components. On the other

hand, this increased connectivity and heterogeneous workloads demand distinct Quality of Service (QoS) levels to satisfy their application requirements[4], [5], [6]. These developments have led to building hyper-scale data centres and complex multi-tier computing infrastructures that require new innovative approaches in managing resources efficiently and provide reliable services. Deployment of 5G and related infrastructures like dynamic network slicing for high bandwidth, high throughput, and low latency applications has only increased the challenges.

Resource Management Systems (RMS) in DCS's are middleware platforms that perform different tasks such as resource provisioning, monitoring, workload scheduling, and many others. Building an efficient RMS for the present and imminent distributed systems are challenging due to many reasons. Significantly, the new class of applications, networks, and Cyber-Physical-Systems (CPS) such as data centres are enormously complex and challenging to fine-tune their parameters manually. For example, "Just 10 pieces of equipment, each with 10 settings, would have 10 to the 10th power, or 10 billion, possible configurations — a set of possibilities far beyond the ability of anyone to test for real" [7], [8]. The emerging network technologies, including 5G and satellite networks, such as Amazon's Project Kuiper and SpaceX's StarLink, have opened up new dimensions [9] and opportunities for developing advanced applications that require high bandwidth, high availability, and low latency. The availability of massive data and advancement in computing capabilities has witnessed the resurgence of Artificial intelligence (AI) techniques driving innovation across different domains such as healthcare, autonomous driving, and robotics [9], [10]. Training AI models itself consumes vast resources and is increasing exponentially and doubling every 3.4 months for the largest AI models (compared to Moores' Law' 2-year doubling period) [7]. The Cloud and Edge infrastructures deliver resources (compute, network, storage) required to accommodate these rapid changes across different domains managed by third-party service providers. These are highly distributed, large-scale, and contain numerous heterogeneous resources. Furthermore, they are multi-tenant, with users sharing the underlying resources with diverse workload characteristics. Thus, providing the performance requirements in such a shared

environment and increasing resource utilisation is a critical problem and challenging for RMS [11].

The existing RMS techniques from operating systems to large scale DCS's are predominantly designed and built using preset threshold-based rules or heuristics. These solutions are static and often employ reactive solutions [12]; they work well in the general case but cannot adjust to the dynamic contexts [13]. Moreover, once deployed, they considerably fail to adapt and improve themselves in the runtime. In complex dynamic environments (such as Cloud and Edge), they are incapable of capturing the infrastructure and workload complexities and hence fall through. Consequently, the AI-centric approaches built on actual data and measurements collected from respective DCS environments are more promising, perform better, and adapt to dynamic contexts. Unlike heuristics, these are data-driven models built based on historical data. Accordingly, AI-centric methods can employ proactive measures by foreseeing the potential outcome based on current conditions. For instance, a static heuristic solution for scaling the resource uses workload and system load parameters to trigger the scaling mechanism. However, this reactive scaling diminishes the users' experience for a certain period (due to the time required for system bootup and application trigger). Consequently, an AI-centric RMS enabled by data-driven Machine Learning (ML) model can predict the future workload demand and scale up or scale down the resources beforehand as needed. Such techniques are highly valuable for both users to obtain better QoS and service providers to offer reliable services and retain their business competency in the market. Moreover, methods like Reinforcement Learning (RL) [13], [14] can improve RMS's decisions and policies by using monitoring and feedback data in runtime, responding to the current demand, workload, and underlying system status.

AI-centric RMS in DCS is more feasible now than ever for multiple reasons. First, AI techniques have matured and have proven efficient in many critical domains such as computer vision, natural language processing, healthcare applications, and autonomous vehicles. Second, most DCS platforms generate enormous amounts of data currently pushed as logs for debugging purposes or failure-cause explorations. For example, Cyber-Physical-Systems (CPS) in data centres already have hundreds of onboard CPU and external sensors monitoring workload, energy, temperature, and weather parameters. Such data is useful to build ML models cost-effectively. Finally, the increasing scale in computing infrastructure and its complexities require automated resource management systems that can deliver the decisions based on the data and key-insights from experience, to which AI models are ideal.

In this regard, this paper makes the following key contributions: (1) presents evolution, and the state-of-the-art RMS techniques in DCS, (2) enlists the challenges associated with data-driven RMS methods, (3) identifies the future research directions and point out the different tasks in which AI-centric methods can be efficiently applied and benefited from, (4) proposes a conceptual data-driven RMS model, and (5) demonstrates two real-time use-cases using data-driven AI methods (related to energy-efficient GPU clock configurations and management of resources in data centres).

The rest of the paper is organised as follows. Section II gives an overview of DCS evolution and state-of-the-art practices in RMS. Section III identifies the challenges associated with data-driven methods. Section IV draws Future research directions. In Section V, a conceptual AI-centric RMS model is presented, and Section VI demonstrates the feasibility of AI-centric methods using two real-time use cases. Finally, the conclusion is drawn in Section VII.

II. DCS EVOLUTION AND THE STATE-OF-THE-ART

An overview of the evolution of primary DCS's is given in Figure 1. Early DCS systems are prominently used in scientific domain applications composed of parallel tasks (distributed jobs in grid computing) and executed on clusters or supercomputing systems. The development of technologies such as service-orientated computing (Web services, REST, SOAP, etc.), virtualisation, and demand for utility-oriented services created the current Cloud computing-based data centres. However, the next decade of DCS's will be driven by IoT-based applications and scenarios that need to process the enormous amount of data and derive meaningful intelligence and business values from it. These IoT-based applications consist of numerous sensors and computing nodes distributed across different network layers from Edge to remote Cloud. Thus, requiring an autonomic sense-connect-actuate model [1] where application tasks are composed, deployed, and executed autonomously—demanding additional machine-to-machine interactions (compared to the current human-to-machine interactions). RMS should autonomously provision resources, schedule application tasks, and manage their demand for QoS and low latency.

In parallel to system advancements, application models have continued to evolve and create new software design patterns like micro-services and execution models like serverless or Function as Service (FaaS) computing. To that end, managing these modern resources and applications requires intelligent decisions enabled from the AI-centric solutions. Although AI-centric RMS techniques will be applicable for all the computing paradigms discussed here, we mainly keep our discussions and illustrations around the Cloud and Edge computing paradigms.

With the increased scale and complexities in next-generation DCSs, traditional static or heuristics solutions are becoming inadequate. These methods require careful hand pruning and human intervention to adapt to the dynamic environments [13]. Consequently, AI-centric data-driven solutions are promising, and there have been many attempts in recent years to address the resource management problems using the data-driven ML solutions [12]. For example, Google has achieved a 40% efficiency in managing its cooling infrastructure using simple ML techniques and learning from historical data [15]. Many other methods explored problems such as device placement, scheduling, and application scaling using data-driven methods [16], [17]. At the system architecture level, [18] used massive

	Cluster/ supercomputers	Cloud	Edge cloud+ IoT
Timeline	1990-2000	2005-2018	2020~2030
Scale (# Nodes)	~10,000	~Millions	~Billions
Features	• Homogenous nodes connected by LAN • To perform similar tasks in parallel	• Geographically distributed heterogenous nodes by WAN, CDNs • To effectively use resources, delivered as utilities	• Geographically distributed large scale heterogenous nodes + sensors, connected by WAN, wireless 5g, satellite, etc • To autonomously sense, connect and actuate applications
Application Models	• Multiple processes running on clusters • MPI, OpenMP. Distributed objects, Web Services (WSDL, UDDI) in case of Grids	• Monolithic applications in isolated Virtual environments (Virtual Machines) • Web 2.0(SOAP, REST)	• Stateless microservices in containers • Lightweight application components interacting with RESTful APIs, Serverless/ FaaS
Application domains & e.g., systems	• Scientific Applications, SETI@Home • Top500 supercomputers	• Enterprise business, personal, social media • Private clouds, Public Clouds (Azure, AWS, Google)	• Enterprise business, personal applications • Smart XX (Industry 4.0), smart city, smart grid, etc,.

Fig. 1: An overview of contemporary DCS evolution (Timeline shows approximate time of the genesis of the system and evolved as mainstream with some overlapping's. The points shown for all dimensions are representative but not exhaustive and only lists the important facets.)

data sets of hardware performance counters and profiles collected from large-scale Google data centre servers and utilised this data to reason, analyse and mitigate front-end stalls in warehouse-scale systems. However, data-driven AI solutions for RMS are in its superficial stage. They require meticulous attention to address the challenges they pose and simultaneously identify potential avenues to incorporate these methods. Moreover, it is essential to build the general frameworks and standards to adopt AI solutions in resource management that are scalable and manageable.

III. CHALLENGES

In this section, we identify and describe the critical issues associated with the adoption of AI solutions in the resource management of distributed computing systems.

A. Availability of Data

The quality of data used to train the models determines the success of machine learning techniques. Also, this data should be available in large quantities with enough features covering all the aspects of environments [19], [20]. Within DCS, multiple challenges exist concerning the availability of such data. First, currently, different resource abstraction platforms collect the data at different granularity. The physical machine-level data from onboard sensors and counters is gathered and accessed by tools like Intelligent Platform Management Interface (IPMI), while at a higher abstraction level, middleware platforms collect data related to workload level, user information, and surrounding environmental conditions (temperature, cooling energy in the data centre). Also, network elements such as SDN controllers collect data related to network load, traffic, and routing. Unifying these data together and preprocessing it in a meaningful way is a complex and tedious task. The respective tools gather the data in a different format without common standards between them. Hence, building data-pipelines combining various subsystems data is crucial for the flexible adoption of ML

solutions. Secondly, current monitoring systems collect data and push them into logging repositories to be used later for debugging. However, converting this data for ML-ready requires monotonous data-engineering. Hence, future systems should be explicitly designed to gather information that can be directly fed to the ML models with minimal data engineering and preprocessing effort. Lastly, although several publicly available datasets provide workload traces, there are hardly any public datasets available representing various infrastructure, including physical resource configurations, energy footprints, and several other important parameters (due to privacy and NDAs). Therefore, getting access to such data is a challenge and needs collaborative efforts and data management standards from the relevant stakeholders. Moreover, requiring standardised data formats and domain-specific frameworks [21].

B. Managing the Deployment of Models

Training ML models and inference in runtime needs an expensive amount of computational resources. However, one significant challenge is to manage the life cycle of ML models, including deciding how much to train, where to deploy the training modules in multi-tier computing architectures like Edge/Fog. As resources have limited capabilities at a lower level and should be allocated to needful applications, if these scarce resources are predominantly used to train models or run the RL agents, the latency-sensitive applications will experience resource starvation. On the other hand, if the models (RL agents) are trained or deployed in resource enriched cloud, the latency to push the inference decisions or the runtime feedback data to edge nodes shoots up, thus creating a delay-bottlenecks in RMS decisions. Furthermore, ML models tend to learn too much with the expense of massive computational resources. Therefore, the innovative solutions are needed to decide how much learning is sufficient based on specific constraints (resource budget, time-budget, etc.) and estimate context-aware adaptive accuracy thresholds of ML models [22]. To overcome this, techniques like transfer learning, distributed learning can be applied to reduce computational demands [20]. In addition, dedicated CPUs, GPUs, and domain-specific accelerators like Google TPU, Intel Habana, and FPGAs (Azure) can carry out the inference.

C. Non-Deterministic Outputs

Unlike statistical models, which are analogous for its deterministic outputs, ML models are intrinsically exploratory and depend on stochasticity for many of its operations, thus producing the non-deterministic results. For example, the cognitive neural nets, which are basic building blocks for many regressions, classification, and Deep Learning (DL) algorithms primarily rely on the principles of stochasticity for different operations (stochastic gradient descent, exploration phase in RL). When run multiple times with the same inputs, they tend to approximate the results and produce different outputs [10]. This may pose a severe challenge in the DCS, such as Edge and Clouds, where strict Service Level Agreements (SLAs) govern the delivery of services requiring deterministic

results. For example, if a service provider fixes a price based on certain conditions using ML models, consumers expect the price to be similar in all the time under similar settings. However, ML models may have a deviation in pricing due to stochasticity creating the transparency issues between users and service providers. Many recent works have focused on this issue and introduced techniques such as induced constraints in neural nets to produce the deterministic outputs [23]. Yet, stochasticity in the ML model is inherent and requires careful monitoring and control over its output.

D. Black Box Decision Making

The ML models' decision-making process follows a completely black-box approach and fails to provide satisfactory justification for its decisions. The inherent probabilistic architectures and enormous complexities within ML models make it hard to evade the black-box decisions. It becomes more crucial in an environment such as DCS, where users expect valid feedback and explanation for any action taken by the service provider. This is instrumental in building trust between service providers and consumers. For instance, in case of a high overload condition, it is usual that service provider shall preempt few resources from certain users with the expense of certain SLA violations. However, choosing which users' resources should be preempted is crucial in business-driven environments. This requires simultaneously providing fair decisions and valid reasons. Many works have undertaken to build the explanatory ML models (Explainable AI- XAI) to address this issue [24], [25]. However, solving this continues to remain a challenging task.

E. Lightweight and Meaningful Semantics

The DCS environment having heterogeneous resources across the multi-tiers accommodates different application services. RMS should interact between different resources, entities, and application services to efficiently manage the resources. However, these requisites semantic models that represent all these various entities meaningfully. Existing semantic models are either heavy or inadequate for such complex environments. Therefore, lightweight semantic models are needed to represent the resource, entities, applications, and services without introducing the overhead [26].

F. Complex Network Architectures, Overlays, Upcoming Features

Network architectures across DCS and telecom networks are evolving rapidly using software-defined infrastructure, hierarchical overlay networks, Network Function Virtualization (NFV), and Virtual Network Functions (VNF). Commercial clouds like Amazon, Google, and Microsoft have recently partnered with telecom operators worldwide to deploy ultra-low latency infrastructure (AWS Wavelength and Azure Edge Zone, for example) for emerging 5G networks. The explosion of data from these 5G deployments and resource provisioning for high bandwidth, throughput, and low latency response

through dynamic network slicing requires a complex orchestration of network functions [27].

In future DCS, RMS needs to consider these complex network architectures, the overlap between telecom and public/private clouds, and service function orchestration to meet end-to-end bandwidth, throughput, and latency requirements. These architectures and implementations, in turn, generate enormous amounts of data at different levels of the hierarchical network architecture. As different types of data are generated in different abstraction levels, standardised well-agreed upon data formats and models for each aspect needs to be developed.

G. Performance, Efficiency, and Domain Expertise

Many ML algorithms and RL algorithms face performance issues like a cold-start problem. Specifically, RL algorithms spend a vast amount of the initial phase in exploration before reaching its optimal policies creating an inefficient period where the decisions are suboptimal, even completely random or incorrect leading to massive SLA violations [20]. RL-based approaches also face several challenges in the real world including (1) need for learning on the real system from limited samples, (2) safety constraints that should never or at least rarely be violated, (3) need of reward functions that are unspecified, multi-objective, or risk-sensitive, (4) inference that must happen in real-time at the control frequency of the system [28]. In addition, AI models are compute-heavy and designed with a primary focus on accuracy-optimisation resulting in a massive amount of energy consumption [7]. Consequently, new approaches are needed to balance the trade-offs between accuracy, energy, and performance overhead. Furthermore, current ML algorithms, including neural network architectures/libraries, are primarily designed to solve computer vision problems. Adapting them to RMS tasks needs some degree of transformation of the way input and outputs are interpreted. Currently, many AI-centric RMS algorithms transform their problem space and further use simple heuristics to interpret the result back and apply to the RMS problems. Such complexities demand expertise from many related domains. Thus, newer approaches, algorithms, standardised frameworks, and domain-specific AI frameworks are required to adopt AI in RMS efficiently.

IV. Future Research Directions

Despite the challenges associated, AI solutions provide many opportunities to incorporate these techniques into RMS and benefit from them. In this section, we explore different avenues where AI techniques can be applied to manage distributed systems resources.

A. Data-driven Resource Provisioning and Scheduling

Resource provisioning and scheduling are a fundamental element of an RMS. Usually, resources are virtualised, and specifically, computing resources are delivered as Virtual machine (VM) or lightweight containers. The problems related to provisioning such as estimating the number of resources required for an application, co-locating workloads based on

their resource consumption behaviours and several others can be addressed using AI techniques. These techniques can be extended to special case provisions such as spot instances. Utilising spot instances for application execution needs careful estimation of application run time (to avoid the state corruption or loss of computation if resources are preempted) and accordingly deciding resource quantity and checkpointing logic. It may require building prediction models based on previous execution performance counters or correlating with clusters based on existing knowledge base [29].

In edge computing environments, RMS should utilise resources from multi-tier infrastructure, and selecting nodes from different layers also requires intelligence and adaptation to application demands and infrastructure status. Furthermore, data-driven AI solutions can be used in scheduling tasks such as finding an efficient node, VM consolidation, migration, etc. The prediction models' historical data and adaptive RL models can be used to manage dynamic scheduling and resource provisioning.

B. Managing Elasticity using Predictive Analytics

Elasticity is an essential feature providing flexibility by scaling up or scaling down the resources based on the applications' QoS requirements and budget constraints. Current approaches in elasticity are based on the reactive methods where resources are scaled according to the system load (in terms of the number of users and input requests). However, such reactive measures diminish the SLAs due to bootup time and swift burst loads. In contrast, forecasting the future load based on the application's past usage behaviours and proactively scaling the resources beforehand vastly improves SLAs and saves costs. Essentially, it needs time series analysis to predict future load using methods like ARIMA or more advanced RNN techniques such as LSTM networks that are proven to be efficient in capturing the temporal behaviours [30]. Such proactive measures from service providers enable efficient management of demand response without compromising the SLAs.

C. Energy Efficiency and Carbon footprint Management

One of the major challenges of computing in recent years has been energy consumption. Increasing reliance on computing resources has created enormous energy, economic and environmental issues. It is estimated that by 2025, data centres itself would consume around 20% of global electricity and emit up to 5% of the world's carbon emissions [31]. Energy efficiency can be achieved across the computing stack from managing hardware circuits to data centre level workload management. Recent studies have shown promising results of AI techniques in device energy-optimised frequency management [29], intelligent and energy-efficient workload management (scheduling, consolidation), reducing cooling energy by fine-tuning cooling parameters [15], [32], and executing applications within power budgets [12], etc. In addition, it can also be effectively used in minimising the carbon-footprints by forecasting renewable energy and shifting the workloads

across clouds accordingly. Each of these subproblems can be addressed by using a combination of predictive and RL models based on application scenarios and requirements.

D. Security and Privacy Management

As cybersystems have become sophisticated and widely interconnected, preserving the privacy of data and securing resources from external threats has become quintessential. Dealing with security has the implications far beyond resource management, including privacy-preserving and complying with the respective jurisdiction's rules. For instance, RMS with user-level schedulers can classify input records and process records with privacy sensitivity within local resource environments (e.g., private cloud) and others on public clouds. One such work is carried out by the University of Washington [33] wherein a deep learning method is used to classify medical records into sensitive and nonsensitive based on data privacy. They created a user-level scheduler for the Aneka Cloud application platform and able to process sensitive medical records on their private cloud and nonsensitive records on Amazon AWS EC2 public cloud.

If resources are maliciously compromised, RMS should adapt to the requirements of the security concerns. There has been widespread use of ML algorithms in many aspects of security management. It includes AI-based Intruder Detection Systems (IDS) to prevent unauthorized access, anomaly detection [34], [35] to identify the deviations in the application/ resource behaviors. AI techniques, including Artificial Neural Networks (ANNs), ensemble learning, Bayesian networks, association rules, and several classification techniques like SVM, can be effectively utilised to address these security-related problems [36]. They can also be predominantly used in preventing Denial-of-service attacks (DDoS) by analysing traffic patterns and filtering suspected traffic, hence, preventing the system failures [37]. Such measures vastly help to manage the resources securely and thus increasing the reliability of the system.

E. Managing Cloud Economics

Cloud economics is a complex problem and requires vast domain knowledge and expertise to price services adequately. It is also essential for consumers to easily understand pricing models and estimate the cost for their deployments. Current pricing models largely depend on subscription types, e.g., reserved, on-demand, or spot instances. The pricing for these subscription models is driven by standard economic principles like auction mechanisms, cost-benefit analysis, profit, revenue maximisation, etc. These pricing problems are solved using techniques like Operation Research (OR) or stochastic game theory approaches [38]. However, such methods are mostly inflexible, and they either overprice the services or results in loss of revenues for cloud service providers. In this regard, ML models can forecast resource demand, and accordingly, excessive resources can be pooled in the open market for consumers. In addition, pricing can be more dynamic based on this forecasted demand response that benefits both consumers and service providers.

F. Generating the Large-scale Data Sets

Machine learning models require large amounts of training data for improved accuracy. However, access to large scale data is limited due to privacy and lack of capabilities to generate a large quantity of data from infrastructure. To that end, AI models itself can be used to create large-scale synthetic datasets that closely depict the real-world datasets. For instance, given a small quantity of data as input, Generative Adversarial Networks (GANs) can be used to produce large-scale data [39]. Such methods are highly feasible in generating time-series data of DCS infrastructure. Moreover, these methods can also be leveraged to produce datasets from the incomplete datasets adequately. Such large-scale data sets are necessary to train efficient predictive models and bootstrap the RL agents to achieve a reasonable efficiency in its policies.

G. Future System Architectures

Cloud services have recently undergone a shift from monolithic applications to microservices, with hundreds or thousands of loosely-coupled microservices comprising the end-to-end application. In [4], the authors explore the implications of these microservices on hardware and system architectures, bottlenecks therein, and lessons for future data centre server design. Microservices affect the computation to communication ratio, as communication dominates, and the amount of computation per microservice decreases. Similarly, microservices require revisiting whether big or small servers are preferable. In [18], the authors use an always-on, fleet-wide monitoring system, to track front-end stalls, I-cache and D-cache miss (as cloud microservices do not lend them amenable to cache locality unlike traditional workloads) across hundreds and thousands of servers across Google's warehouse-scale computers. The enormous amounts of data generated and analysed help provide valuable feedback for the design of next-generation servers. Similarly, deep learning can be used to diagnose unpredictable performance in cloud systems. Data from such systems can thus be invaluable for the hardware and system architectures of future DCS.

H. Other Avenues

Along with the aforementioned directions, AI-centric solutions can be applied to several other RMS tasks, including optimising the heuristics itself [19], network optimisations (e.g., TCP window size, SDN routing optimisation problems), and storage infrastructure management [20]. Moreover, learning-based systems can be extended across different computing system stack, from lower abstraction levels, including hardware design, compiler optimisations, operating system policies, to a higher level interconnected distributed system platforms[13].

V. Conceptual Model for AI-centric RMS

In the AI-centric RMS (Resource Management Systems) system, models need to be trained and deployed for the inference used by the RMS for different tasks. However, integrating

data-driven models into DCS platforms in a scalable and generic manner is challenging and is still at a conception stage. In this regard, as shown in Figure 2, we provide a high-level architectural model for such data-driven RMS. The essential elements of this system are explained below. It consists of three entities:

Users/ Applications: Users requiring computing resources or services interact with the middleware using APIs or interfaces.

AI-centric RMS Middleware: This is responsible for performing different tasks related to managing user requests and underlying infrastructure. The AI-centric RMS tasks continuously interact with the data-driven models for accurate and efficient decisions. The RMS needs to perform various tasks, including provisioning the resources, scheduling them on appropriate nodes, monitoring in runtime, dynamic optimisations like migrations, and consolidations [12] to avoid the potential SLA violations. Traditionally, these tasks are done using the algorithms implemented within the RMS system that would execute the policies based on the heuristics or threshold-based policies. However, in this AI-centric RMS, the individual RMS operations are aided with inputs from the data-driven models. The data-driven AI models are broadly categorised into two types, (1) predictive models and (2) adaptive RL models. In the former, models are trained offline using supervised or unsupervised ML algorithms utilising historical data collected from the DCS environment that includes features from resources, entities, and application services. This data is stored in databases, and data-engineering is done, such as preprocessing, cleaning, normalising, to suit AI models' requirements. Thus, this offline training can be done on remote cloud nodes to benefit from the specialised, powerful computing resources. The trained models can be deployed on specialised inference devices like Google Edge TPU and Intel Habana. Choosing the optimal place and deciding where to deploy these ML models depends on where the RMS engine is deployed in the environment, and this is itself a challenging research topic that should be addressed as described in Section III-B.

In the latter case, runtime adaptive models such as Reinforcement Learning (RL) that continue to improve their policies based on agents' interactions and system feedback. It requires both initial learning and runtime policy improvement methods that need to be updated after every episode (certain time reaching to terminal state). The RMS operations can interact with both the predictive and RL-based data-driven models using the RESTful APIs in runtime [12].

DCS Infrastructure: The computing infrastructure comprises heterogeneous resources, including sensors, gateway servers, edge data centres, and remote clouds. Therefore, adopting the data-driven AI-centric RMS models needs a significant change in the way current RMS systems are designed and implemented, as well as monitoring agents, interfaces, and deployment policies that can be easily integrated into existing environments.
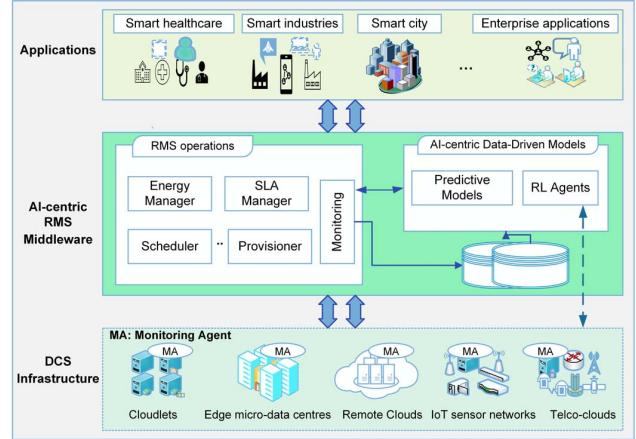

Fig. 2: Conceptual Data-Driven RMS Model
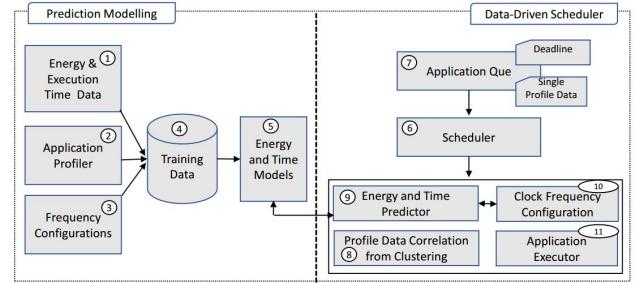
Fig. 3: System Model

VI. DEMONSTRATION CASE STUDIES

In this section, we present two use cases that have applied AI techniques to two different problems: (1) data-driven configuration of device frequencies for energy-efficient workload scheduling in cloud GPUs, (2) data center resource management using ML models.

A. Data-Driven GPU Clock Configuration and Deadline-aware Scheduling

Graphics Processing Units (GPUs) have become the de-facto computing platform for advanced compute-intensive applications such as video processing and autonomous cars. Additionally, ML models are massively reliant on the GPUs for training due to their efficient SIMD architectures that are highly suitable for parallel computations. However, the energy consumption of GPUs is a critical problem. Dynamic Voltage Frequency Scaling (DVFS) is a widely used technique to reduce the dynamic power of GPUs. Yet, configuring the optimal clock frequency for essential performance requirements is a non-trivial task due to the complex nonlinear relationship between the application's runtime performance characteristics, energy, and execution time. It becomes even more challenging when different applications behave distinctively with similar clock settings. Simple analytical solutions and standard GPU frequency scaling heuristics fail to capture these intricacies and
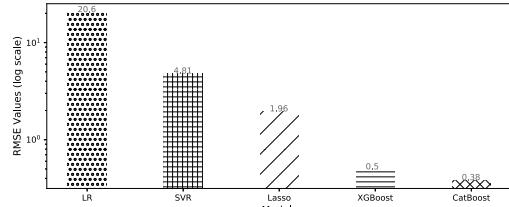
(a) Energy prediction
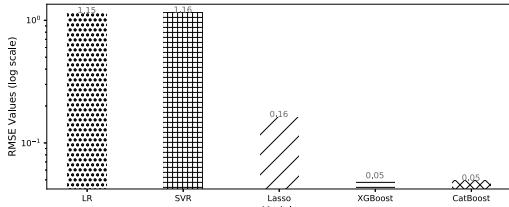
(b) Time prediction

Fig. 4: Performance of different models for energy and execution time prediction (lower RMSE value is preferred)

Fig. 5: Average energy consumption of applications

Fig. 6: Average total energy consumption of GPU

scale the frequencies appropriately. In this regard, we propose a data-driven frequency scaling technique by predicting the power and execution time of a given application over different clock settings. Furthermore, using this frequency scaling by prediction models, we present a deadline-aware application scheduling algorithm to reduce energy consumption while simultaneously meeting their deadlines.

The high-level overview of the system is given in Fig. 3. It is broadly classified into two parts, predictive modelling, and data-driven scheduler. In the first part, we collect the training data that consists of three parts, profiling information, energy-time measurements, and respective frequency configurations. We then predict two entities for a given application and frequency configuration, i.e., energy consumption and execution time. Subsequently, in the second part, the new applications arrive with the dead-line requirements and minimal profiling data from a default clock frequency execution. The scheduler finds correlated application data using the clustering technique, and this data is used for predicting the energy and execution time over all frequencies. Finally, based on the deadline requirements and energy efficiency, the scheduler scales the frequencies and executes the applications. We use twelve applications for evaluation from two standard GPU benchmarking suites, Rodinia and Polybench. The training data is generated from profiling the applications using nvprof, a standard profiling tool from NVIDIA. We collected around 120 key features representing key architectural, power, and performance counters. To build the predictive models, we explored several regression-based ML models, including Linear Regression (LR), lasso-linear regression (Lasso), and Support Vector Regression (SVR). Also, ensemble-based gradient boosting techniques, extreme Gradient Boosting (XGBoost),
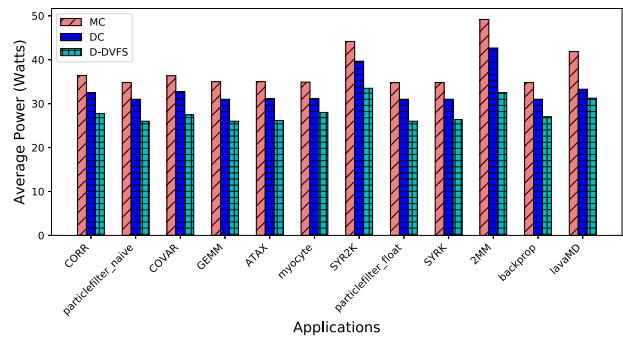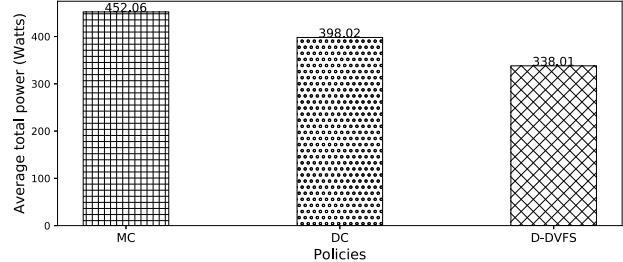
and CatBoost. The goal is to build energy and execution time prediction models for each GPU device to assist the frequency configuration.

We conduct extensive experiments on NVIDIA GPUs (TESLA P100). The experiment results have shown that our prediction models with CatBoost have high accuracy with the average Root Mean Square Error (RMSE) values of 0.38 and 0.05 for energy and time prediction, respectively (Figure 4a, Figure 4b). Also, the scheduling algorithm consumes 15.07% less energy (Figure 6) as compared to the baseline policies (default and max clock) while meeting the application deadlines as our approach can scale the frequencies that have energy-efficient settings (Figure 7) also able to meet performance requirements. More details on prediction-models, scheduling
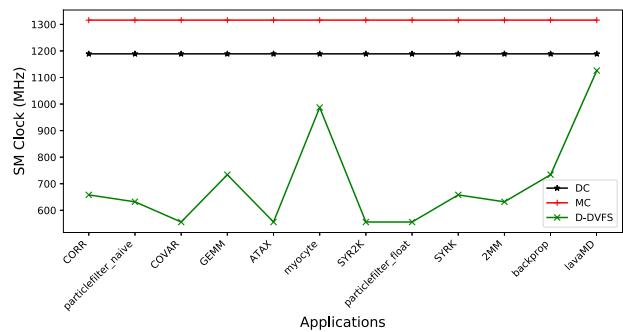

Fig. 7: Frequency Scaling by different policies

8
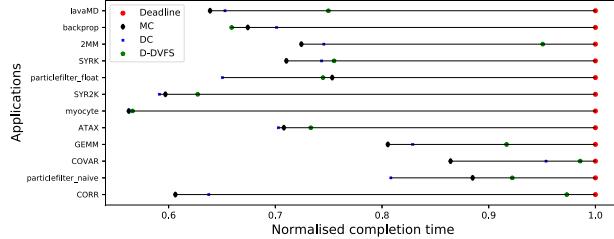
Fig. 8: Normalised application completion time compared to deadline

algorithms, and implementation can be found in [29].

B. Industrial (Google Cloud and Microsoft Azure) Data Center Management

Data centres are the backbone infrastructures of cloud computing today. A data centre is a complex Cyber-Physical-System (CPS) consists of numerous elements. It houses thousands of rack-mounted physical servers, networking equipment, sensors monitoring server, and room temperature, a cooling system to maintain acceptable room temperature, and many facility-related subsystems. The data centre is one of the highest power density CPS of up to 20 kW per rack, dissipating an enormous amount of heat. This poses a serious challenge to manage resources energy efficiently and provide reliable services to users. Optimising data centre operation requires tuning the hundreds of parameters belonging to different subsystems where heuristics or static solutions fail to yield a better result. Moreover, even a 1% improvement in data centre efficiency leads to savings in millions of dollars over a year and also helps to reduce the carbon footprints. Therefore, optimising these data centres using potential AI techniques is of great importance. Accordingly, we discuss two real-time AI-based RMS systems built by researchers at Google and Microsoft Azure Cloud.

ML-centric cloud [12] is an ML-based RMS system at an inception stage from the Microsoft Azure cloud. They built Resource Control (RC)—a general ML and prediction serving system that provides the insights of workload and infrastructure for re-source manager of Azure compute fabric. The input data collected from the virtual machine and physical servers. The models are trained using a gradient boosting tree and trained to predict the different outcomes for user's VMs such as average CPU utilisation, deployment size, lifetime, and blackout time. The Azure resource manager interacts with these models in runtime. For instance, the scheduler queries for virtual machine lifetime, and based on the predicted value; the appropriate decision is taken to increase infrastructure efficiency. Applying these models to several other resource management tasks is under consideration, including power management inside Azure infrastructure.

Similarly, Google has also applied ML techniques to optimise the efficiency of their data centres. Specifically, they have used ML models to change the different knobs of the cooling

system, thus saving a significant amount of energy [15]. The ML models are built using simple neural networks and trained to improve the PUEs (Pow-er Usage Effectiveness), a standard metric to measure the data centre efficiency. The input features include total IT workload level, network load, parameters affecting the cooling system like outside temperature, wind speed, number of active chillers, and others. The cooling subsystems are configured according to the predictions, and results have shown that the 40% savings are achieved in terms of energy consumption. Therefore, the brief uses cases presented here strongly attest to the feasibility of AI-centric solutions in different aspects of resource management of distributed systems.

VII. Conclusions

Future distributed computing platforms will be massively complex, large scale, and heterogeneous, enabling the development of highly connected resource-intensive business, scientific, and personal applications. Managing resources in such infrastructure require data-driven AI approaches that derive key insights from the data, learn from the environments, and take resource management decisions accordingly. In this paper, we have discussed the challenges associated with the adoption of AI-centric solutions in RMS. We identified the potential future directions describing different RMS tasks where we can apply AI techniques. Moreover, we presented the conceptual AI-centric RMS model. Finally, we demonstrated the two use-cases of AI-Centric solutions in resource management of distributed systems.

The state-of-the-art rule-based or heuristics resource management solutions have become inadequate in modern distributed computing platforms. The RMS policies need to deal with massive scale, heterogeneity, and varying workload requirements. As a result, we believe that AI techniques and tools can be widely utilised in numerous RMS tasks, including monitoring, resource provisioning, scheduling, and many others. Such approaches are highly adaptive and better suited to deal with the resource management complexities, enabling optimised resource management from processor to middleware platforms, and application management.

References

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[2] Norton, "The future of iot: 10 predictions about the internet of things," *https://us.norton.com/internetsecurity-iot-5-predictions-for-the-future-of-iot.html*, 2019.

[3] I. Baldini, P. Castro, K. Chang, P. Cheng, S. Fink, V. Ishakian, N. Mitchell, V. Muthusamy, R. Rabbah, A. Slominski *et al.*, "Serverless computing: Current trends and open problems," in *Research Advances in Cloud Computing*. Springer, 2017, pp. 1–20.

[4] Y. Gan, Y. Zhang, D. Cheng, A. Shetty, P. Rathi, N. Katarki, A. Bruno, J. Hu, B. Ritchken, B. Jackson *et al.*, "An open-source benchmark suite for microservices and their hardware-software implications for cloud & edge systems," in *Proceedings of the 24th International Conference on Architectural Support for Programming Languages and Operating Systems*, 2019, pp. 3–18.

[5] A. V. Dastjerdi and R. Buyya, "Fog computing: Helping the internet of things realize its potential," *Computer*, vol. 49, no. 8, pp. 112–116, 2016.

[6] A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica *et al.*, "Above the clouds: A berkeley view of cloud computing," *Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS*, vol. 28, no. 13, p. 2009, 2009.

[7] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, "Green ai," *arXiv preprint arXiv:1907.10597*, 2019.

[8] D. Amodei and D. Hernandez, "Ai and compute," *https://blog.openai.com/ai-and-compute*, 2018.

[9] G. Giambene, S. Kota, and P. Pillai, "Satellite-5g integration: A network perspective," *IEEE Network*, vol. 32, no. 5, pp. 25–31, 2018.

[10] S. Russell and P. Norvig, *Artificial intelligence: a modern approach.* Prentice Hall, 2002.

[11] R. Buyya, S. N. Srirama, G. Casale, R. Calheiros, Y. Simmhan, B. Varghese, E. Gelenbe, B. Javadi, L. M. Vaquero, M. A. Netto *et al.*, "A manifesto for future generation cloud computing: Research directions for the next decade," *ACM Computing Surveys (CSUR)*, vol. 51, no. 5, pp. 1–38, 2018.

[12] R. Bianchini, M. Fontoura, E. Cortez, A. Bonde, A. Muzio, A.-M. Constantin, T. Moscibroda, G. Magalhaes, G. Bablani, and M. Russinovich, "Toward ml-centric cloud platforms," *Communications of the ACM*, vol. 63, no. 2, pp. 50–59, 2020.

[13] J. Dean, "Machine learning for systems and systems for machine learning," in *Presentation at 2017 Conference on Neural Information Processing Systems*, 2017.

[14] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction.* MIT press, 2018.

[15] J. Gao, "Machine learning applications for data center optimization," 2014.

[16] A. Mirhoseini, H. Pham, Q. V. Le, B. Steiner, R. Larsen, Y. Zhou, N. Kumar, M. Norouzi, S. Bengio, and J. Dean, "Device placement optimization with reinforcement learning," in *Proceedings of the 34th International Conference on Machine Learning.* JMLR. org, 2017, pp. 2430–2439.

[17] S. Tuli, S. Ilager, K. Ramamohanarao, and R. Buyya, "Dynamic scheduling for stochastic edge-cloud computing environments using a3c learning and residual recurrent neural networks," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2020.

[18] G. Ayers, N. P. Nagendra, D. I. August, H. K. Cho, S. Kanev, C. Kozyrakis, T. Krishnamurthy, H. Litz, T. Moseley, and P. Ranganathan, "Asmdb: understanding and mitigating front-end stalls in warehouse-scale computers," in *Proceedings of the 46th International Symposium on Computer Architecture*, 2019, pp. 462–473.

[19] C. Cummins, P. Petoumenos, Z. Wang, and H. Leather, "End-to-end deep learning of optimization heuristics," in *Proceedings of the 26th International Conference on Parallel Architectures and Compilation Techniques (PACT).* IEEE, 2017, pp. 219–232.

[20] I. A. Cano, "Optimizing distributed systems using machine learning," Ph.D. dissertation, University of Washington, Seattle, USA, 2019.

[21] I. Portugal, P. Alencar, and D. Cowan, "A survey on domain-specific languages for machine learning in big data," *arXiv preprint arXiv:1602.07637*, 2016.

[22] A. Toma, J. Wenner, J. E. Lenssen, and J.-J. Chen, "Adaptive quality optimization of computer vision tasks in resource-constrained devices using edge computing," in *Proceedings of the 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, 2019, pp. 469–477.

[23] J. Y. Lee, S. V. Mehta, M. Wick, J.-B. Tristan, and J. Carbonell, "Gradient-based inference for networks with output constraints," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 4147–4154.

[24] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins *et al.*, "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," *Information Fusion*, vol. 58, pp. 82–115, 2020.

[25] D. Gunning, "Explainable artificial intelligence (xai)," *Defense Advanced Research Projects Agency (DARPA), nd Web*, vol. 2, 2017.

[26] M. Bermudez-Edo, T. Elsaleh, P. Barnaghi, and K. Taylor, "Iot-lite: a lightweight semantic model for the internet of things and its use with dynamic semantics," *Personal and Ubiquitous Computing*, vol. 21, no. 3, pp. 475–487, 2017.

[27] H. Zhang, N. Liu, X. Chu, K. Long, A.-H. Aghvami, and V. C. Leung, "Network slicing based 5g and future mobile networks: mobility, re-source management, and challenges," *IEEE communications magazine*, vol. 55, no. 8, pp. 138–145, 2017.

[28] G. Dulac-Arnold, D. Mankowitz, and T. Hester, "Challenges of real-world reinforcement learning," *arXiv preprint arXiv:1904.12901*, 2019.

[29] S. Ilager, R. Muralidhar, K. Rammohanrao, and R. Buyya, "A data-driven frequency scaling approach for deadline-aware energy efficient scheduling on graphics processing units (gpus)," in *Proceedings of the 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*, 2020, pp. 579–588.

[30] Y. Gan, Y. Zhang, K. Hu, D. Cheng, Y. He, M. Pancholi, and C. Delimitrou, "Leveraging deep learning to improve performance predictability in cloud microservices with seer," *ACM SIGOPS Operating Systems Review*, vol. 53, no. 1, pp. 34–39, 2019.

[31] J. M. Lima, "Lima. Data centres of the world will consume 1/5 of Earth's power by 2025." 2017. [Online]. Available: https://data-economy.com/data-centres-world-will-consume-1-5-earths-power-2025/

[32] S. Ilager, K. Ramamohanarao, and R. Buyya, "Etas: Energy and thermal-aware dynamic virtual machine consolidation in cloud data center with proactive hotspot mitigation," *Concurrency and Computation: Practice and Experience*, vol. 31, no. 17, p. e5221, 2019.

[33] X. Xu and X. Zhao, "A framework for privacy-aware computing on hybrid clouds with mixed-sensitivity data," in *Proceedings of the 17th International Conference on High Performance Computing and Communications*, 2015, pp. 1344–1349.

[34] S. K. Moghaddam, R. Buyya, and K. Ramamohanarao, "Acas: An anomaly-based cause aware auto-scaling framework for clouds," *Journal of Parallel and Distributed Computing*, vol. 126, pp. 107–120, 2019.

[35] I. Butun, B. Kantarci, and M. Erol-Kantarci, "Anomaly detection and privacy preservation in cloud-centric internet of things," in *Proceedings of the International Conference on Communication Workshop (ICCW).* IEEE, 2015, pp. 2610–2615.

[36] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2015.

[37] X. Yuan, C. Li, and X. Li, "Deepdefense: identifying ddos attack via deep learning," in *2017 IEEE International Conference on Smart Computing (SMARTCOMP).* IEEE, 2017, pp. 1–8.

[38] S. Mistry, A. Bouguettaya, H. Dong *et al.*, *Economic Models for Managing Cloud Services.* Springer, 2018.

[39] C. Zhang, S. R. Kuppannagari, R. Kannan, and V. K. Prasanna, "Generative adversarial network for synthetic time series data generation in smart grids," in *Proceedings of the International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm).* IEEE, 2018, pp. 1–6.