2019 IEEE International Conference on Service-Oriented System Engineering (SOSE)

Improving Data Center Efficiency Through Holistic Scheduling In Kubernetes

Paul Townend[1], Stephen Clement[1], Dan Burdett[1], Renyu Yang[1], Joe Shaw[1], Brad Slater[1], Jie Xu[12]

[1]Edgetic Ltd., 1 City Square, Leeds, LS1 2ES, UK
[2]School of Computing, University of Leeds, LS2 9JT

{ *paul.townend, stephen.clement, dan.burdett, renyu.yang, joe.shaw, brad.slater, jie.xu* } @ edgetic.com

***Abstract*— Data centers are the infrastructure that underpins modern distributed service-oriented systems. They are complex systems-of-systems, with many interacting elements, that consume vast amounts of power. Demand for such facilities is growing rapidly, leading to significant global environmental impact. The data center industry has conducted much research into efficiency improvements, but this has mostly been at the physical infrastructure level. Research into software-based solutions for improving efficiency is greatly needed. However, most current research does not take a holistic view of the data center that considers virtual and physical infrastructures as well as business process. This is crucial if a solution is to be applied in a realistic setting. This paper describes the complex, system-of-systems nature of data centers, and discusses the service models used in the industry. We describe a holistic scheduling system that replaces the default scheduler in the Kubernetes container system, taking into account both software and hardware models. We discuss the initial results of deploying this scheme in a real data center, where power consumption reductions of 10-20% were observed. We show that by introducing hardware modelling into a software-based solution, an intelligent scheduler can make significant improvements in data center efficiency. We conclude by looking at some of the future work that needs to be performed in this area.**

Keywords- kubernetes, docker, containers, energy, power, efficiency, data center, scheduling, behavior, modeling, analytics, holistic, PUE, interference

I. INTRODUCTION

In recent years there has been a huge increase in the use and proliferation of distributed service-oriented applications. This has been driven by many factors, including increasing numbers of users, IoT devices, streaming audio and video services, cloud storage, and the rising popularity of microservices. Furthermore, growth is predicted to accelerate as driverless vehicles [12], connected devices, and smart infrastructure become increasingly common [13][14].

These applications are typically hosted in data centers – large facilities containing compute, storage, and network resources. Data centers are digital factories that process electrical power into digital services and generate enormous quantities of waste heat that requires additional power to remove. They are complex systems of systems, with many interacting elements.

The data center industry currently consumes 2-3% of global electrical power, with global data traffic more than doubling every four years [15]. At the same time, the utilization of any one data center is typically estimated to be between 10-20% [1][2]. As the size and number of data centers increases, their environmental impact (such as power consumption and carbon emissions) and the impact of their relative inefficiency becomes increasingly important.

Traditionally, the data center industry has addressed inefficiencies primarily through hardware innovation – more efficient CPUs, better cooling solutions, improved power distribution systems, etc. However, there is huge scope to offer better efficiencies through software-only solutions – examples include more powerful monitoring and analysis of data center systems [16], predictive capacity planning [17], CFD-based floor layout tools [18], and intelligent scheduling mechanisms [19]. One common factor that is necessary when developing real-world solutions in this space is the need for multi-disciplinary, holistic thinking – it is not enough to view such a system in isolation; instead solutions need to consider the interactions between the physical systems, software systems, business processes, and user behaviors.

In this paper, we look in detail at the complex nature of the data center infrastructure behind service-oriented systems and discuss a number of the challenges faced by academia and the industry. We discuss the need for improved metric collection and monitoring, the need for better mathematical models of hardware behavior, and the potential for better scheduling mechanisms within data centers. We conclude by describing in detail two case studies where improved scheduling mechanisms were developed using advanced hardware modelling to significantly reduce power consumption in a realistic setting.

II. DATA CENTERS

Although data centers are the fundamental infrastructure on which service-oriented computing is based, to date there has been relatively little work [3][4][5][6] in the literature that considers them in a truly holistic manner – with hardware systems influenced by software and vice versa. Such an approach is crucial if theoretical research is to be translated into practical, real-world technology.

A. Data centers as systems-of-systems

A data center is a complex, interacting system of systems, each of which is typically the concern of a different

978-1-7281-1442-2/19/$31.00 ©2019 IEEE
DOI 10.1109/SOSE.2019.00030

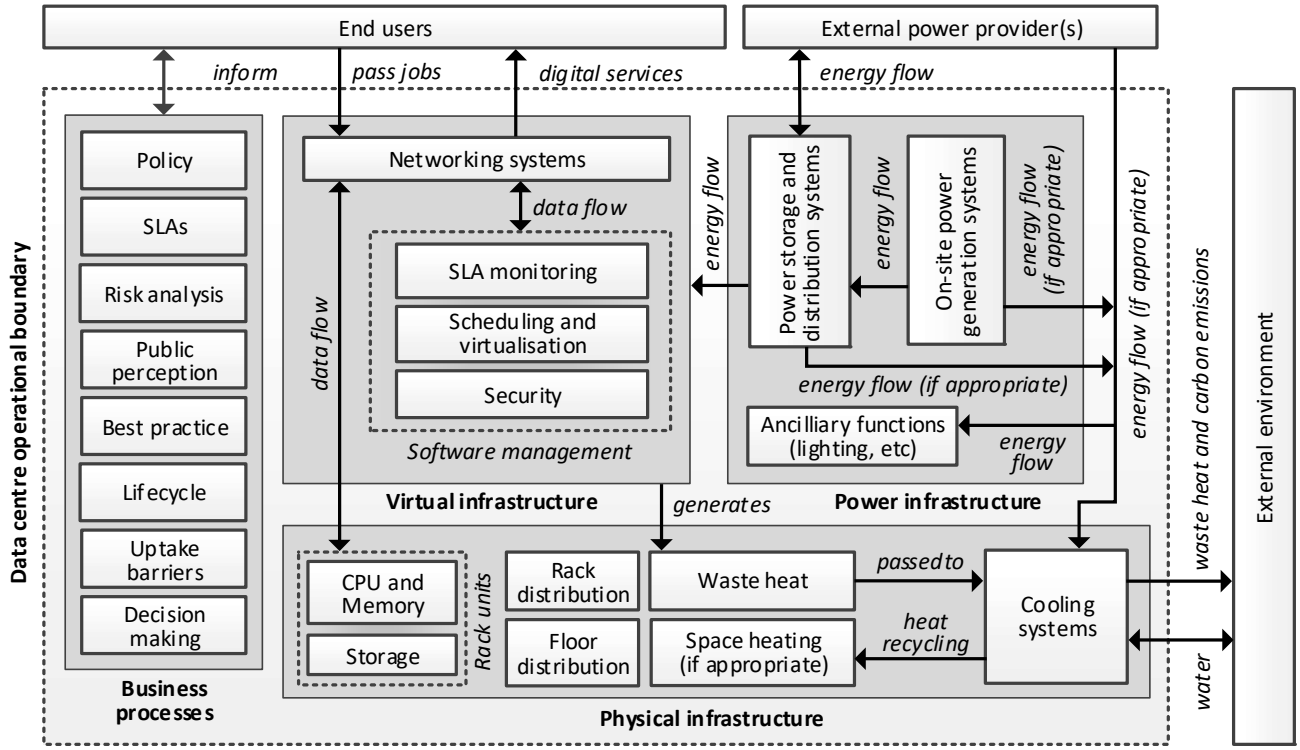

Figure 1. A Data Center as a complex system of systems

discipline (computer science, mechanical engineering, electrical engineering, etc.) Figure 1 identifies these interactions at a high-level; from this view, the components of a data center can be categorized into:

1) Physical infrastructure. The fundamental hardware components that process data in a data center are *servers*. Servers consist of a variety of components, the most important of which are: CPU, Disk, Memory, and Network interfaces. They consume electrical power and generate heat. Servers are typically stored in *racks*, which are enclosures consisting of multiple servers (typically with a network switch). Racks are sometimes organised into *pods*, and arranged into aisles to improve the efficiency of cooling systems. Cooling systems typically (in the case of "air cooling") take heated air and through a variety of mechanisms, cool this down before recirculating it. Some systems use liquid cooling to a variety of degrees. Cooling systems themselves consume a significant amount of electrical power [7].

2) Power infrastructure. The physical infrasturcture of a data center consumes significant amounts of power. In addition to the power requirements of servers, physical infrastructure needs include conversion from AC to DC supply, backup generation systems, distribution systems, and ancilliary functions such as lighting. Electrical power may cost different amounts at different times of day, offering opportunities for intelligent scheduling of non-urgent software jobs; this will increasingly be a factor when Smart Grid technology becomes more widespread.

3) Virtual infrastructure. This category is comprised of all software systems running on the physical infrastructure of the data center. This includes supervisory systems (often known as DCIM – Data Center Infrastructure Management) that monitor the physical infrastructure to ensure that services are available, that service-level agreements (SLAs) are maintained, and that no anomalous events have occurred. Virtualisation software is typically also present; this serves to create, manage, migrate and remove the co-located containers and virtual machines that contain user jobs, and also to ensure that security constraints (particularly integrity and confidentiality) are maintained.

4) Business processes. Any solution proposed to alter the working environment of a data center needs to be compatible with the business processes and requirements of that system. These include the (often variable) service-level agreements offered to users, the decision making process (holistic solutions can often struggle as budgets for facililty and operations may be siloed, creating difficulty in determining at which level the solution sits), and the public perception of the data center.

These complex, interacting systems consume extremely large amounts of electrical power. Although data centers currently consume approximately 2-3% of the global

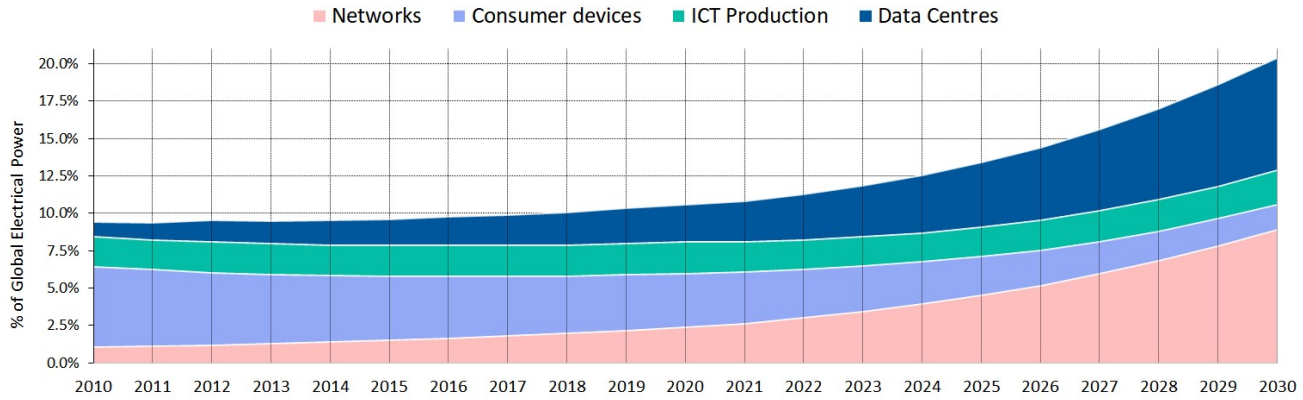

Figure 2. Expected global electrical power usage (%) 2010-2030

electrical supply, there is much speculation that this figure will rise significantly in the coming decade. We now look at some of the trends in data center energy consumption, as well as the metrics used to gauge efficiency in the industry.

B. Energy usage of data centers and PUE

Figure 2, based on [15] shows an "expected case" projection of energy usage for networks, consumer devices, ICT production, and data centers. This predicts that data center energy use will consume approximately 8% of global electrical power by 2030 (approximately 8000 TWh.) It is notable that in the same work, the "worst case" estimate predicts that data centers will consume 20% of the global electrical supply by 2030.

Currently, the data center industry measure energy efficiency primarily through the use of a single metric – PUE (Power Usage Effectiveness.) PUE was first proposed by [8] and is the ratio of total facility energy used by a data center to the energy delivered to computing equipment (e.g. servers). Any resource in a data center that consumes energy but is not a computational device is considered to contribute to facility energy. PUE is therefore expressed as:

$$\text{PUE} = \frac{\text{Total Facility Energy}}{\text{IT Equipment Energy}}$$

The largest consumer of facility energy is typically the cooling systems within a data center; as racks of servers produce waste heat, mechanisms are needed to cool this down – often at considerable energy cost. A recent survey of over 1200 data centers [9] showed that respondents, on average, had a PUE of 1.89 – in other words, for every 1 watt of energy used to power IT equipment, another 0.89 watts was spent on facility energy (cooling, power distribution, etc.) Separately, a recent technical report by the European Commission [10] showed an average surveyed data center PUE in the EU of 1.64. The industry generally perceives that the lower a data center's PUE score is, the more efficient that data center is. This has been reflected by a sustained series of innovations in the industry that has sought to continually reduce PUE; figure 3 (also [10]) shows the downward trend in average PUE values for surveyed data centers between 2009-2016.

It is notable that PUE is usually heavily affected by the external environment of a data center; the warmer the location of a data center, the higher the PUE (primarily due to the need for more powerful cooling.) As efficiency improvements at a facility level slow due to diminishing returns as PUE gets closer to its minimum value of 1.00 (at which point all facility power would be devoted to IT equipment), research into IT equipment energy reduction becomes increasingly important.

However, there are significant problems with the PUE metric: for example, a single metric such as PUE cannot differentiate between different carbon footprints (e.g. use of renewable or non-renewable energy), water consumptions, etc. Most notably, *data center innovations which seek to improve efficiency through reductions in IT energy consumption may actually cause a PUE score to become worse (higher)* – if total facility energy remains at a similar level, but through better resource management IT equipment energy consumption can be reduced, PUE will increase. **This poses a significant barrier to the adoption of software-based operational efficiency solutions in the current climate**. This is a consideration that must be addressed by

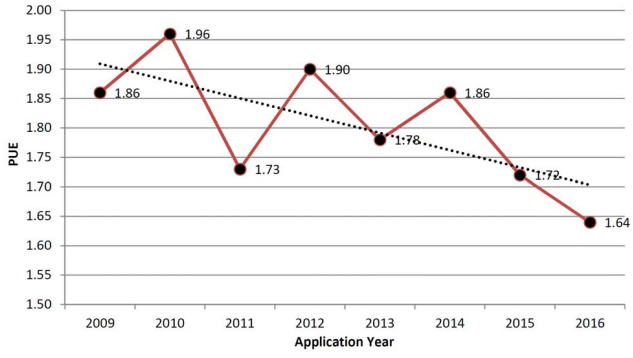

Figure 3. Long-term downward trend in average data center PUE [10]

future research – possibly through the introduction of improved efficiency metrics, etc.

Given the increasing importance of reductions in IT equipment energy consumption within a data center, we discuss in section 4 a case study into how software solutions - utilizing a holistic approach to all aspects of a data center - can offer potentially large reductions in power consumption.

III. Data Center Types

Data centers come in many different types and categories, which can be differentiated by size, ownership, and service model. The category of a data center significantly affects the type of software-based efficiency solutions that can be placed on them.

A major distinction is between *private* (only accessible to those within a single organisation and sometimes selected partners) and *public* (accessible by many parties, who may not be known to the data center owner.) Additionally, data centers may be purposed in different ways. For example:

- ***Co-locational*** data centers require customers to supply their own IT hardware. This hardware is placed in the facility and provided with physical security, cooling, and power, but the data center provider has no control over the servers themselves (either the software running on them or their power status, BIOS status, etc.)
- ***Telecoms*** data centers are facilities used by telecommunications providers; these typically feature very high connectivity requirements, running specialised software services.
- ***Dedicated-hosting*** data centers provide server capacity to single customers, with no sharing of machines (this is sometimes known as "bare metal", or MaaS.) Again, these data centers typically do not have any control over the software running on the servers themselves (although this can vary, depending on the facility).
- ***Managed hosting*** data centers provide servers and storage systems for customers, typically with a platform-as-a-service (PaaS) or software-as-a-service (SaaS) model. Services offered may include database storage, web hosting, systems monitoring, etc. Servers may be owned by the customer or the data center, and the extent of control provided to each party can vary significantly.
- ***Shared hosting*** data centers (of which *cloud* data centers are an example) typically provide virtualised and multi-tenant resources for multiple customers to use. Such data centers typically provide user interfaces that allow customers to automatically deploy virtualised resources; the underlying physical location of these resources is determined by the data center's scheduling software.
- ***Edge***. Edge data centers are typically smaller than traditional data centers and are located closer to where data is generated – thus reducing delay and cost in network transmission. Edge is considered a significant growth area, driven by Internet-of-Things, wireless sensor networks, streaming video uptake, etc.
- ***Hybrid***. A data center facility containing more than one of the above service models.

There is an increasing trend in the data center industry for the software running on physical servers to be virtualised and co-hosted (and thus, a trend towards shared hosting data centers). **This offers a significant opportunity for software-based efficiency solutions**, as the resource management within such systems is centralised, providing realistic opportunities for metric collection and analysis, and improved scheduling mechanisms.

A. Virtualised resources

From a user perspective, it is preferable to develop applications and services that are agnostic to the underlying system or geographic location. Additionally, users demand configurable deployments of services, incorporating all required resources, that are able to scale to demand. Virtualisation solves this for computing by packing applications alongside required libraries, for networking by allowing reconfigurable software defined networks (SDN) and for storage by allowing users to expand storage with their requirements.

From the perspective of a data center, virtualisation increases the value of individual servers by allowing multiple users to be seamlessly collocated on a single physical machine (PM). Virtualisation allows sharing of the PM while ensuring isolation between users without any complicated configuration requirements from the users – thus greatly increasing capacity (i.e. the ability to take on far more users than there are physical machines). Indeed, deployment and migration between machines can become an automated process, greatly reducing management overhead. Furthermore, a failure of a virtualised resource is contained and will (or at least should) not affect the running or operation of any other virtual resource located on the same physical host.

The two primary types of virtualised computing resource used in modern data centers are virtual machines and containers (operating system-level virtualisation). Virtual machines are packages that include an operating system, set of applications, and associated data. They function on top of a hypervisor, which is located upon a base operating system and both emulates a PM and provides isolation for each VM.

Containers are a similar concept but remove the need for a hypervisor: a container image includes a set of applications and library dependencies, occasionally alongside user data but does not contain an operating system. Multiple containers are hosted on a single operating system with the operating system itself providing isolation between each container. A container can be deployed on any machine that

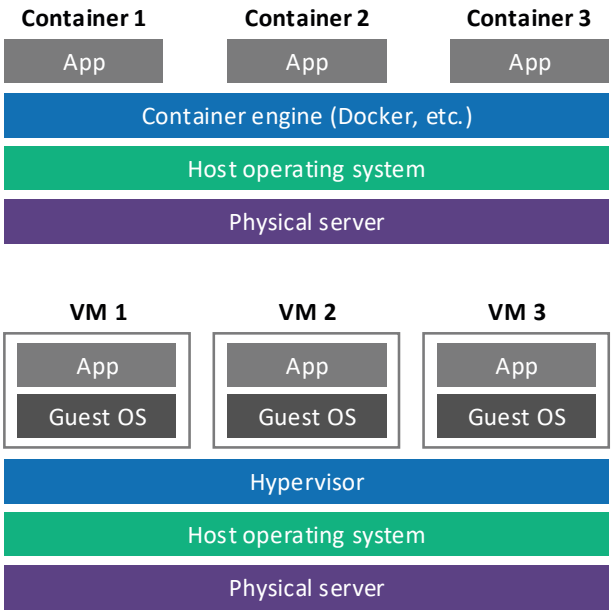

Figure 4. Comparison of Containers and Virtual Machines

runs the operating system it targets. The exclusion of an OS provides a significant reduction in image size and thus means that containers can be created, deployed, and destroyed in a much faster manner. Furthermore, since the host OS manages containers the density (the number of containers capable of running simultaneously) can be far greater than with virtual machines where the hypervisor often requires a 1-to-1 mapping from physical to virtual resources. Figure 4 shows a high-level architectural comparison of containers and virtual machines.

Within the cloud community, there has been a significant movement to container-based technology. Figure 5 shows a normalised comparison of google searches between the term "virtual machine" and "docker" (the most popular container type, chosen for this comparison to reduce semantic uncertainty in the search result) between 2014-2019. Although this is only an indication of interest, the trend is marked and significant. Interest in containers has been fuelled by rising use of microservice-based architectures, which are becoming increasingly common, especially in connected sensor-based systems such as those found in internet-of-things solutions [11].

IV. A Holistic Solution for Improving Efficiency

As has been discussed, there is not only a great need but also great potential for software-based solutions for improving efficiency within data centers. It is important to consider that different types of data center require different approaches, which need to take a holistic view of the data center (including at least its physical and virtual infrastructures) if they are to be applicative in real-world scenarios. As an example of the effectiveness of a focused, holistic software innovation, we now present a solution we have developed that targets shared hosting data centers. Specifically, we assume:

- The target data center will be a shared hosting facility that uses virtualized resources
- The virtualized resources will be packaged within Docker containers
- We can gain access to detailed metrics with regard the data center environment (specifically: air temperature) and servers.
- Kubernetes is used as the underlying container orchestration system.

Kubernetes (*https://kubernetes.io/*) is an open-source container orchestration system, originally developed by Google and now maintained by the Cloud Native Computing Foundation. Although the Docker system (*https://www.docker.com/*) is becoming the de-facto standard for hosting software containers, there are a number of competing systems that automatically scale, manage, update and remove containers across distributed resources. Kubernetes is the market-leader and the standardized means of orchestrating containers and deploying distributed applications; uptake is growing rapidly across many organisations, and increasing numbers of data centers are using it to build distributed container infrastructures.

A. *Intelligent scheduling in Kubernetes*

By default, the Kubernetes system attempts to load-balance all the nodes (physical servers) in the cluster that it manages by spreading containers evenly across it. This makes logical sense when considering the system from the perspective of its virtual infrastructure only, as it reduces the likelihood of individual nodes becoming unduly resource

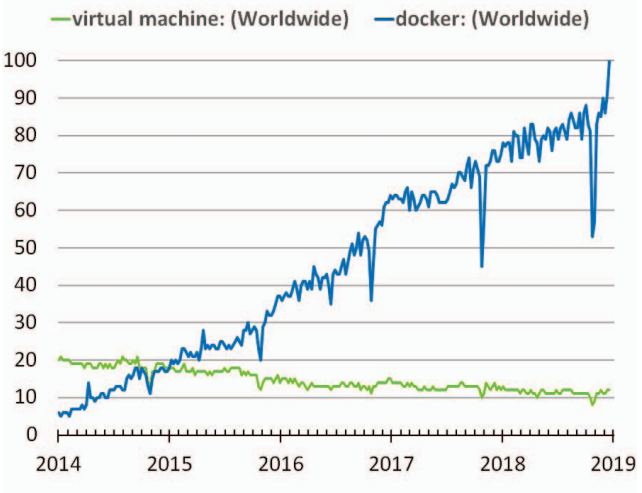

Figure 5. Google searches: virtual machines and containers (Docker) between 2014-2019

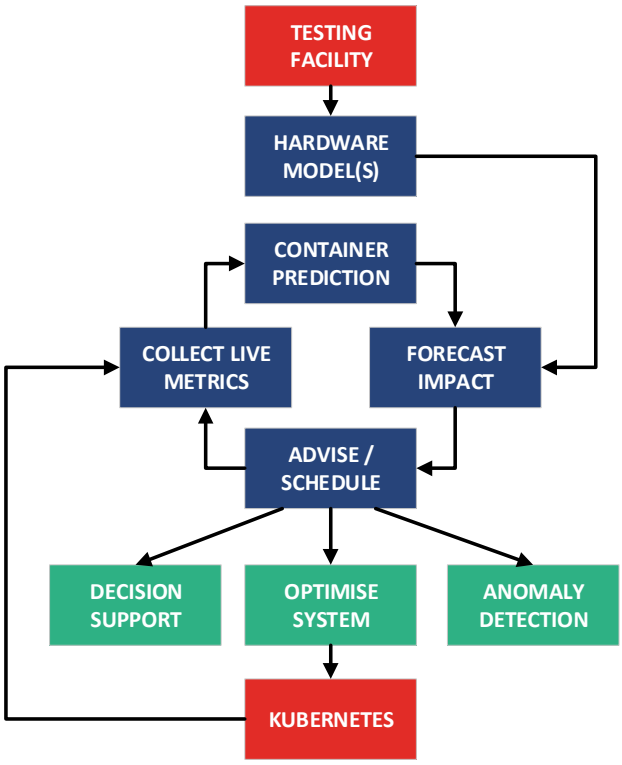

Figure 6. High-level architecture of the Edgetic solution

constrained while others are relatively lightly loaded. In reality, the underlying physical hardware in a cluster will react to load (in other words – resource utilization - particularly of the CPU, but also disk, memory, and network transfer) in a wide variety of ways. Different levels of hardware load will cause differing amounts of waste heat to be generated, and different amounts power to be consumed, depending on the type of server, the BIOS settings of that server, and even the ambient temperature of the server location.

We propose that with a significantly better understanding of the hardware response of each server within a data center, and a predictive capability to assess the impact of each incoming container within a system, an intelligent scheduling mechanism can be created whereby containers are allocated to nodes in a more optimal fashion – tuned to either reducing power consumption, increasing performance (instructions-per-cycle), or to a balance in-between.

To achieve this, we need to establish detailed server models, predictive container behaviour models, and detailed knowledge of the environment of the data center. Each of these is a complex task and requires a different solution. Figure 6 shows the broad strategy to our scheme; hardware behaviour to a range of different loadings is modelled at a test facility. This is used to forecast the impact (in terms of performance and power consumption) of container placement within the target cluster, the outcome of which is used to optimally place a container within the cluster. Simultaneously, live metrics are collected from the system and fed into container prediction models – using historical and online learning to more accurately predict the behaviour of each incoming software container (i.e. what resources it will use at what time period, etc.)

B. Hardware modelling

One of the benefits of modelling server hardware behaviour is that it allows us to treat a data center (or a cluster of machines) as an optimisation problem whereby the goal, be it to decrease power consumption or increase performance, can be mathematically or analytically solved. This is true whether a data center is made up of multiple different kinds of servers or a homogenous set of servers.

To achieve this, it is necessary to experimentally determine the link between multiple attributes - including thermal environmental, power consumption, and internally logged server metrics – before condensing these down into a form of multivariate polynomial equation that can be acted upon in machine learning.

To achieve this, we placed a server (specifically a Dell R430) into an instrumented wind tunnel (see figure 7). We controlled and monitored the thermal environment of the server while running computational benchmarks within the server to test components such as CPU, memory, disk, and network. These tests satisfy a design of experiments of varying complexities, depending on time availability and accuracy requirements.

One of the biggest challenges we faced is being able to properly explore the design space that is required to model the behaviour of a server without dedicating an unreasonable amount of time to testing. This is in part because such a complicated system requires multiple variables to describe it, and each additional variable has an exponential impact on the size of the design space; additionally, the relationship between the variables and the model is almost always of an order higher than one, and each increase in order is a further increase in the size of the design space. The issue is further compounded by the amount of time required for each individual test to achieve a steady state for one or more of the variables. For this reason, a full factorial exploration of such a space is unfeasible, and so specific Designs of Experiments are employed that sample the space at required points to capture the order of behaviour previously determined experimentally, plus interactions and a margin to allow for the unexpected.

The process of creating these models is being continually updated. Contemporary models include metrics that were only added when previous models were found to be missing a variable that could describe their full behaviour at the time. The counterpoint to this is that models existing currently may also include variables that are in practice not statistically significant in modelling behaviour for a certain server, or even all servers in general. The evolution of this understanding is an ongoing process.

Figure 7. Wind tunnel used to generate server hardware behaviour models

C. *Software metric collection*

In addition to hardware metric collection, it is also necessary to collect a large number of metrics from the containers running across a distributed cluster; through analysis of these metrics, predictions can be made about the behaviour of newly submitted container jobs, which combined with an appropriate hardware model, can inform a more intelligent scheduling solution. However, this is not a simple task – not only does the collection of metrics impose a significant overhead itself on each node within a cluster, but the amount of data generated for analysis can be extremely large. There are many research challenges to be addressed when deciding what metrics are of most significance to predictive algorithms, and at what granularity these should be collected (as often as required to give an acceptable level of accuracy, but not so often as to impose too much overhead in terms of either storage or analysis).

At first, creating a metric collection system seems to be a simple task: many system metric collectors exist and are "pluggable" in the sense that they could attach various extra components and publish data into various storage systems. However, this is not as simple as it first appears. An important requirement to any metric collection system is to have a very low resource footprint (ideally no more than 5% of a CPU core), and little memory impact. Being able to store metrics in an appropriate database for analysis is also important.

There are a host of metric collection systems for Kubernetes. A number of vendor specific solutions exist, such as *Datadog*, and open source solutions, such as *Elastic Stack* with *Metricbeat* (*https://elastic.co/products/beats/metricbeat*). When considering a generic Kubernetes setup consisting, typically, of Docker engines running containers and using Linux *cgroups* for resource accounting and constraints, the host level metric collectors are not as informed as required. Ideally, a system needs to monitor the host resources as well as those for individual Kubernetes specific cgroups to provide both details of machine utilisation and potential container contention.

Initially we looked at the native "out-the-box" solution, Heapster. Now discontinued, Heapster pulls node specific metrics from the Kubelets (Kubernetes node agent) at regular intervals and can push these into a limited number of databases. We used Heapster, Kafka and Kafka stream processors initially to perform distributed power model calculations for the scheduling component. However, it was clear that the Heapster model would not scale, causing metric lag. The time to detect newly started pods (collections of containers) was poor, leading to poorer scheduling decisions.

Making the predictions and collecting metrics needed to happen as close to the source as possible, before providing scheduling scores. Rather than cannibalise an existing system to do this, or set up a further complex processing pipeline, we made a decision to create our own metric collector.

The Edgetic metric collector is written in the Go programming language, with a very simple set of abstractions allowing additions of collector types and publisher types to be added/removed when required. The metric agent runs on every node, monitoring node stats, cgroups, performance events, etc. We can run models generated from benchmarking directly within the agent.

Once the metric collector is in place, the next step is to create a storage infrastructure. This is easy to achieve when benchmarking a few servers with a timeseries database publisher or similar; however, when storing metrics for a large cluster, we need to leverage a queuing system system (such as Kafka) to deal with back pressure and network latency, before shipping the metrics into long term storage.

We use open source technology (such as Fluentd) to store high resolution metrics, featuring hundreds of data points per cycle depending on how many pods are running, at relatively high frequency. Metric data compresses well, due to its structured nature, and can be stored in time order for later retrieval. As an example, a week of compressed metric logs during one set of our experiments was approximately 800GB compressed, and roughly 8TB when decompressed.

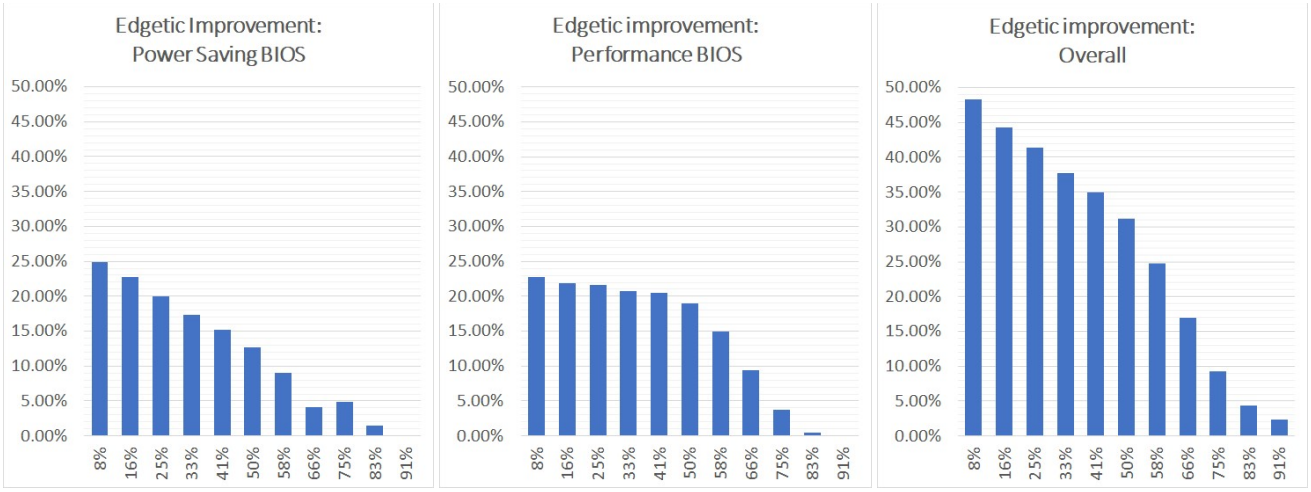

Figure 8. Energy efficiency improvement: a) Edgetic vs. Kubernetes default scheduler in power-saving mode; b) Edgetic vs. Kubernetes default scheduler in performance mode; c) Edgetic scheduler in power-saving mode vs, Kubernetes default scheduler in performance mode

Exposing this data for processing is a different matter. In-cluster databases such as Elasticsearch or Prometheus can be used for short term visibility and alerting, but the indexes will quickly become very large, requiring frequent clean-up. Long term analysis of data is a significant challenge; it is possible that Spark or similar Map-reduce systems on top of distributed storage may be most effective.

V. RESULTS

To test the effectiveness of our holistic approach, we implemented the system described above and tested it in two stages. Both stages of experimentation were performed in a real data center: RISE SICS North in Lulea, Sweden.

A. Holistic scheduler experiment I

We applied our initial solution to a cluster of 56 Dell R430 servers running Kubernetes, where server behaviour was modelled using the wind tunnel and metric collection methods above. As an example of how server settings change the nature of behaviour within a data center, we modelled and tested the servers using two distinct BIOS settings: "performance mode" whereby the CPU and fans run at full speed irrespective of server utilisation (in other words, maximum performance and power consumption at all times), and "power saving mode" whereby the CPU is scaled up using DVFS (dynamic voltage frequency scaling) dependant on load – this setting offers much reduced power, but at the price of slow CPU performance and response at low utilisation.

Workloads of known resource requirements were submitted to the cluster and allocated based on the optimal placement suggested by the hardware models; performance and power consumption were measured and then compared against the same workloads running on the default Kubernetes scheduler. As workloads were deterministic, the performance metrics for our scheduler were measured as the percentage increase/decrease in workload completion time over the default Kubernetes scheduler. Experimental runs were made across both BIOS modes.

Results from the experiment demonstrate potentially huge gains in overall energy efficiency (measured as improvement in performance plus reduction in power consumption – i.e. instructions-per-watt), although this is dependent on overall data center / cluster utilisation. Results shown for different levels of CPU utilisation across both BIOS types are show in Figure 8.

One significant finding is that by using an intelligent scheduler that utilised an accurate hardware model, we could schedule containers onto physical machines in a way that maintained performance even when in the "power saving" BIOS mode. This allows us to make an idealised comparison between the relative performance and efficiency of a data center using the Edgetic scheduler in power-saving mode vs the Kubernetes default scheduler in performance mode (shown as the "Overall" result in Figure 8).

It should be noted that these results are idealised - there is a strong assumption that the workload coming into the data center is deterministic and known in advance; in reality, this will almost never be the case, and so the results from the holistic scheduler I experiments serve primarily as an indication of what may be achieved with sufficiently accurate workload prediction.

Following the success of these experiments, we began to look at not only increasing node scale (to measure the feasibility of our metric collection routines) but also loosening the deterministic workload assumption.

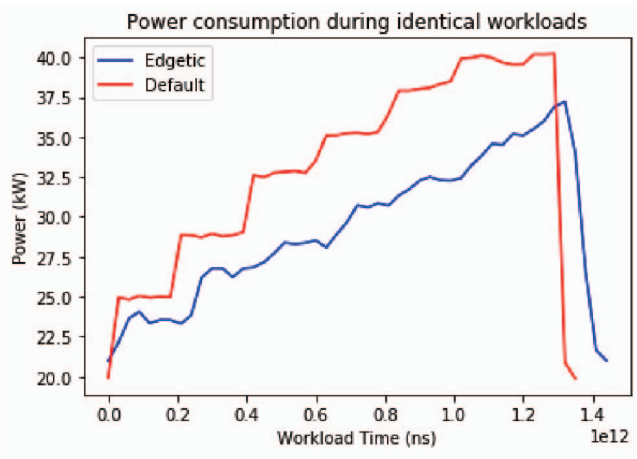

Figure 9. Holistic scheduler experiment II power consumption comparison between the Edgetic scheduler and the Kubernetes default

B. Holistic scheduler experiment II

For our second set of experiments, we targeted a Kubernetes cluster running on 215 machines at RISE SICS North. All the servers in this cluster were made up of OCP (Open Compute Project) hardware – low-cost machines that are often used in large (hyperscale) data centers. We modelled the OCP server in our wind tunnel system, using an improved model creation system based on feedback and analysis from our first set of experiments. Furthermore, we extended our scheduler to make basic predictions on workload based on historical submission. This allowed us to create a range of experimental workloads that could be submitted in a deterministic fashion but did not need to be known in advance by our scheduler.

Based on the findings of our new hardware model, the Edgetic scheduler loaded fewer OCP servers to a higher level (to the optimum level described in the model), keeping the remaining servers in the cluster at idle until required. Over the course of the experiment, an increasing number of synthetic workloads were submitted to the data center in batches that approximately corresponded to 10% total utilisation, from 10% to 70% approximate total utilisation.

The batches were additive, so the 10% batch ran for the whole experimental duration, while the 70% batch only ran for the last period of the workload. The experimental workloads were submitted to the default and Edgetic schedulers in sequence, with a small delay to allow the datacenter to return to an approximate steady-state.

The results of these experiments were extremely encouraging; although performance remained at a similar level on both the Edgetic scheduler and the default Kubernetes scheduler, the Edgetic scheduler lowered the power consumption of the cluster by 10-20% (shown in Figure 9.) This was achieved purely through optimizing container placement based on the hardware model.

Figure 10 shows the distribution of jobs across the server racks for each batch. Here you can see the differing effects of the Edgetic and Kubernetes schedulers. Notably, as the Edgetic scheduler takes into account server inlet temperature in its calculations we can see that certain locations in the datacenter are preferred. These correspond to the cooling outlet which is located closest to servers at index (10,10) in the diagram.

A surprising result was discovered in the power utilisation for each workload; the Edgetic workload consistently uses less power even when the datacenter becomes highly utilized, and the performance advantage of the Kubernetes scheduler starts to disappear.

VI. CONCLUSIONS

Data centers are the fundamental infrastructure on which modern distributed systems are constructed, and lend themselves heavily to cloud and micro-service based systems. However, as the size of data centers increases, their power consumption is becoming an increasingly important part of the global electrical supply, with significant financial and environmental ramifications. Research into software-based solutions for improving efficiency in these systems is greatly needed; however, most existing work considers only virtual infrastructure. To be applicable in the industry – and to maximise effectiveness – it is essential that solutions take a holistic view, considering virtual and physical infrastructure as well as business processes.

We describe the complex, system-of-systems nature of data centers, and provide a category of data center service models; the selection of an appropriate service model is crucial for any research into providing infrastructure management software in the data center space. To illustrate the effectiveness of a holistic approach, we describe a new scheduler we have developed for Kubernetes-based clusters and data centers, that considers both physical and virtual infrastructures. We discuss the server modelling we performed as part of this work, as well as the metric collection tools we developed to model workload behaviour. The developed scheduler was then tested in two experiments performed at the RISE SICS data center in Sweden, on 50 and 215 physical servers respectively.

Our initial results demonstrate power consumption reductions of between 10-20% depending on data center utilisation. This shows that by using hardware modelling alone, an intelligent scheduler can make significant improvements in data center efficiency, even before further optimisations are introduced.

A. Future work

There is much future research to be done in this space. A particularly important strand of research is that of **QoS and performance interference modelling and prediction.** Multi-tenant servers in cloud data centers may run heterogeneous workloads such as latency-critical services (typically encapsulated in Kubernetes containers or virtual machines) alongside other batch-like jobs with lower-priorities. Some jobs in a cluster are allocated resources (i.e., provisioned) to handle maximum loads and unexpected load spikes. As a result, a cluster might remain under-utilized. To enable a fully utilized cluster system, thereby maximizing

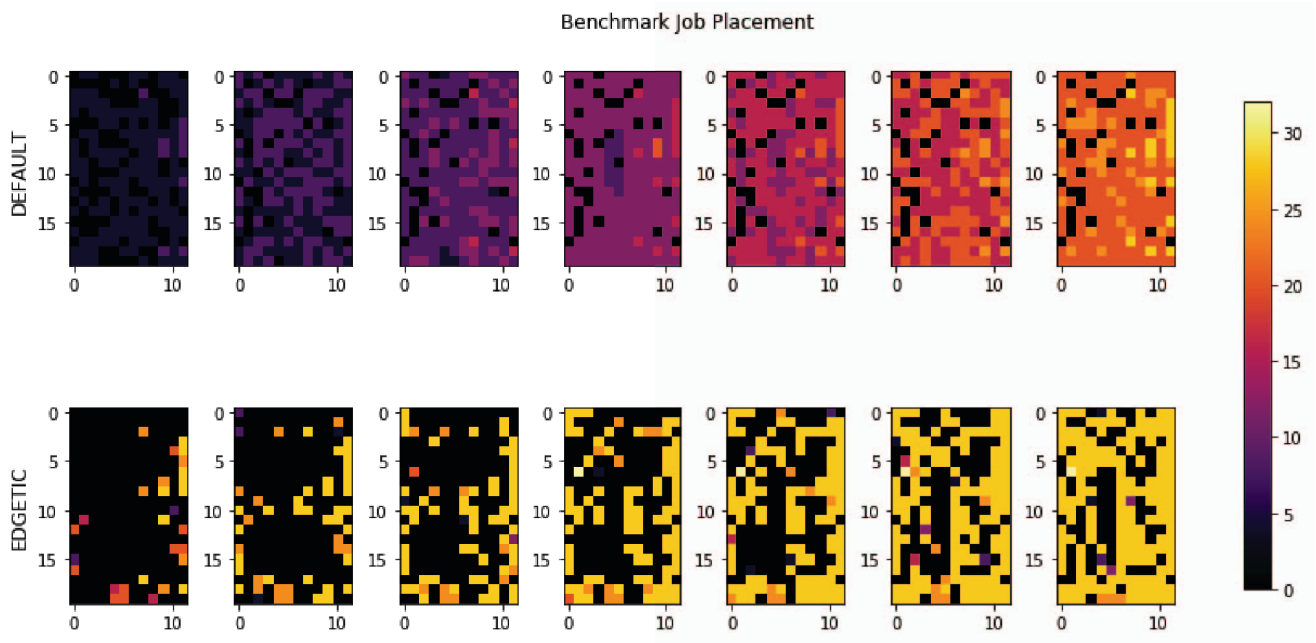

Figure 10. Comparison of cluster server utilization per job batch, using the default Kubernetes scheduler and the Edgetic scheduler

the pay back of capital investment, temporarily unused resources from over-provisioned workloads can be used for running best-effort jobs.

There is a trend whereby increasing numbers of cloud-based stateful applications may overwhelm conventional batch jobs, particularly boosting the requirement for strict QoS guarantees and performance interference throttling. This type of interference can occur on almost any type of shared resource, including CPU, storage, network, etc. A typical way of minimizing the interference is to allocate dedicated cores for applications and avoid oversubscribing the cores. However, different cores may share the same processors, which still leads to a great amount of resource contention. In particular, the last-level cache (LLC) can have a severe impact on the end-user experience, again highlighting the need for better hardware understanding.

To quantify interference, we can deploy fine-grained system counters to monitor and collect the application performance information. Platform-level Performance Monitoring Units (PMU) and performance-metric based early detection can help achieve this goal. For instance, by using kernel-level counters, interference indicators such as average cycles per instruction (CPI), a certain percentile response latency (e.g., 99 percentile tail latency), Last Level Cache (LLC) misses, and especially misses per kilo instructions (MPKI), can greatly help to detect the emergence of performance interference.

Normally it is the case that such metrics have fairly consistent values across running instances and time periods. It is therefore possible to detect outliers from regular behaviour; once the monitored value of one of these counters surpasses the performance threshold of Service Level Objectives (SLO) from a user's perspective, an alert can be used to quickly locate anomalies. Given the precise prediction of interference level, a resource manager can perform the runtime QoS-aware scheduling underpinned by non-invasive performance isolation through fine-grained resource control. An external QoS controller can continuously monitor the resource allocation, resource usage and other environmental variables This has a significant applicability into the resource management of data centers:

- **Resource estimation**. A node agent can leverage a QoS prediction model to calculate how many resources should be reserved in order to meet a specific SLO target. For instance, if a user can tolerate 5% performance degradation, then the QoS target can be slightly reduced to 95%. According to the QoS model, we can roughly estimate a safe resource allocation given the environmental condition.
- **Resource oversubscription**. If we want to target the QoS goal, how many resources should be assigned -- calculate how many resources should be strictly allocated to the pod (regular resource binding as default scheduler did) and how many resources can be spared and borrowed by other low-prioritized jobs/best-effort jobs/other pods (flexible over-subscription).
- **Interference-aware resource sharing**. Furthermore, through detecting the interference producers, we can ideally throttle them rapidly and facilitate the

performance recovery of performance victims. Given this, the system can allow best-effort jobs even if they slightly break the SLO violation of other co-resident workloads.

- **Energy-efficient scheduling with performance considered**. We can integrate the performance interference consideration with the energy efficient scheduling. Although, the QoS-sensitive policy may conflict with the policy towards energy efficiency, we can still leverage the performance indicators to shortlist QoS-safe sub-candidates based on the primary energy efficiency selection.

To better aid holistic approaches in future, there are also significant challenges to be solved to improve the performance overhead of metric collection in data center servers. Current approaches, especially with perf event metrics are slow and resource heavy. Due to this, it may be necessary to look into lower level abstractions such as Linux extended BPF (eBPF) tracing tools so that metrics can be captured in the kernel.

Furthermore, when considering modelling of physical infrastructure, there are concerns about how feasible it is to rely on accurate external sensors being part of the model. Currently a key attribute in each model is the temperature of air going into the server, but this relies on the data center the model is being employed upon to be well instrumented with temperature sensors. Most servers include temperature sensors of varying number and quality and a move to utilising those instead of requiring external sensors would be beneficial. However, issues with this will undoubtably arise from a lack of uniformity regarding internal sensors.

ACKNOWLEDGMENTS

We would like to thank RISE SICS North for their support and the provision of time and servers in their ICE data center. We would also like to thank the University of Leeds for additional computational and storage resources.

REFERENCES

[1] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch, "Heterogeneity and dynamicity of clouds at scale: Google trace analysis," in ACM SoCC, 2012

[2] C. Delimitrou and C. Kozyrakis, "Quasar: resource-efficient and qosaware cluster management," in ACM ASPLOS, 2014

[3] T. Mastelic, I. Brandic. "Recent trends in energy-efficient cloud computing.", IEEE Cloud Computing 2.1 (2015): 40-47.

[4] Y. Joshi, P. Kumar, eds. "Energy efficient thermal management of data centers", in Springer Science & Business Media, 2012

[5] A. Beloglazov, R. Buyya, "Energy efficient resource management in virtualized cloud data centers", in Proc. 10th IEEE/ACM Int. Conf. on cluster, cloud and grid computing (pp. 826-831), May 2010

[6] I. S. Moreno, R. Yang, J. Xu, T. Wo, "Improved energy-efficiency in cloud datacenters with interference-aware virtual machine placement." In Proc. 11th IEEE Int. Symp. on Autonomous decentralized systems, 2013

[7] N. Rasmussen, "Electrical efficiency modeling of data centers.", Schneider Electric White Paper 113 Revision 2, 2011

[8] C. Belady, "The green grid data center efficiency metrics: PUE and DCIE:, Technical report, The Green Grid, Feb. 2007

[9] Super Micro Computer, "Data Centers and The Environment", https://www.supermicro.com/wekeepitgreen/Data_Centers_and_the_Environment_Dec2018_Final.pdf, December 2018

[10] L. Castellazzi, M. Avgerinou, P. Bertoldi, "Trends in data center energy consumption under the European Code of Conduct for data center energy efficiency", European Commission JRE Technical Report, DOI: 10.2760/358256, December 2017

[11] H. Zhu, I. Bayley, "If Docker Is The Answer, What Is The Question?", Proc. 12th Int. Conf. on Service-Oriented System Engineering, Bamberg, March 2018

[12] P. Bansal, K. Kockelman, "Forecasting Americans' long-term adoption of connected and autonomous vehicle technologies", Transportation Research Part A: Policy and Practice, Vol. 95, 2017

[13] B. Silva, M. Khan, K. Han, "Towards sustainable smart cities: A review of trends, architectures, components, and open challenges in smart cities", Sustainable Cities and Society, Vol. 38, 2018

[14] M. H. Cintuglu, O. A. Mohammed, K. Akkaya, A. S. Uluagac, "A Survey on Smart Grid Cyber-Physical System Testbeds," in IEEE Communications Surveys & Tutorials, vol. 19, no. 1, 2017

[15] A. Anders, T. Edler, "On global electricity usage of communication technology: trends to 2030.", Challenges 6, no. 1 (2015): 117-157

[16] J. Gustafsson et al, "A demonstration of monitoring and measuring data centers for energy efficiency using opensource tools" in Proc. 9th Int. Conf. on Future Energy Systems, Karlsruhe, Germany, Jun 2018

[17] F. Kong, X. Liu, "Greenplanning: Optimal energy source selection and capacity planning for green datacenters.", in Proc. 7th IEEE Int. Conf. on Cyber-Physical Systems, 2016

[18] J. Sjölund, M. Vesterlund, N. Delbosc, A. Khan, J. Summers, "Validated Thermal Air Management Simulations of Data Centers Using Remote Graphics Processing Units." In Proc. 44th Annual Conf. of the IEEE Industrial Electronics Society, 2018.

[19] Q. Shaheen et al., "Towards Energy Saving in Computational Clouds: Taxonomy, Review, and Open Challenges," in IEEE Access, vol. 6, 2018.