A Low Carbon Kubernetes Scheduler

Aled James
Email: aledjms@gmail.com

Daniel Schien
University of Bristol, UK
Email: Daniel.Schien@bristol.ac.uk

***Abstract*—A major source of global greenhouse gas emissions is the burning of fossil fuels for the generation of electricity. The portion of electricity generated from fossil fuel varies across regions, and within a region with demand for electricity and the availability of renewable energy sources. Cloud providers operate data centres in locations around the planet. And certain kind of server computation can tolerate migrating between data centres.**

In this paper we describe the design and implementation of a low carbon scheduling policy for the open-source Kubernetes container orchestrator. We apply this scheduler in a form of demand side management by migrating consumption of electric energy to countries with the lowest carbon intensity of electricity.

The primary contributions of this text are (i) the scheduler's design, which provides a generic model for optimising workload placement in regions with the lowest carbon intensity (ii) an evaluation of its performance in a case study with a major public cloud provider (iii) an implementation of a demand side management solution that consumes electricity where, instead of when, grid carbon intensity is lowest.

***Index Terms*—Kubernetes; green computing; DSM; Demand Side Management; renewable energy; grid carbon intensity**

I. Introduction

Cloud datacentres typically comprise tens to thousands of interconnected servers and consumes a substantial amount of electrical energy [1]. [2] estimates that by 2030 datacentres will use anywhere between 3% and 13%[1] of global electricity. All major cloud computing companies acknowledge the need to run their datacentres as efficiently as possible in order to address economic and environmental concerns, and recognise that ICT consumes an increasing amount of energy. As an example for a response, Google Cloud Platform runs its datacentres entirely on renewable energy since the end of 2017 [3], while Microsoft have announced that their global operations have been carbon neutral since 2012 [4]. Not all cloud providers have been able to make such an extensive commitment; Oracle Cloud, for example, is currently 100% carbon neutral in Europe, but not in other regions [5]. Much of the aforementioned companies' claims come with the caveat that their carbon emissions are not zero, but are offset by financial instruments which invest in future renewable energy generation or carbon capture; these future reductions are then netted off against the current year's greenhouse gas emissions [6].

As the availability of renewable energy at a particular location is inherently variable, the electricity in the local grid that datacentres draw from typically is generated from both renewable ('green') energy as well as fossil fuel or nuclear based energy sources ('brown energy') in order to compensate for the intermittent nature of renewable energy generation. Solar photovoltaic (PV) power production primarily depends on the amount of solar irradiation (insolation) reaching the solar panel; however, that irradiation is not uniformly distributed over time [7]. In addition to the rotation of the earth, weather and intermittend clouds block the Sun's rays and thus influence solar power generation output.

Intermittency of availability of renewable energy sources is one of the factors driving demand side management (DSM) in the electric grid where consumers of electric grid alter their energy consumption patterns. In the area of energy systems management, demand side management (contrasts with supply side interventions) refers to any initiatives (technical interventions, pricing models and monetary incentives) that affect how and when electricity is being required by consumers. While much of the research on DSM focusses on domestic energy consumption there has also been work investigating DSM by cloud data centres.

An important form of DSM is load shifting, whereby load on the electric grid (i.e. demand for electric energy) is rescheduled to a time of day during which the energy demand can be more easily met by renewable resources [8]. Fig. 1 provides a basic visualisation of the load shape objective of Load Shifting Demand Side Management.

In this paper we describe the proof of concept design and implementation of a low carbon scheduling policy for the open-source Kubernetes container orchestrator that can provide DSM for cloud data centres. The scheduler selects compute nodes based on the real-time carbon intensity of the electric grid in the region they are in. Real-time APIs that report grid carbon intensity is available for an increasing number of regions; but not exhaustively around the planet. In order to effectively demonstrate the schedulers ability to perform global load balancing we evaluate the scheduler based on its ability to the metric of solar irradiation.

The text is organised as follows. In the next section we look at existing work related to energy consumption of cloud computing. In section III we derive the design of the scheduler – the implementation of which is detailed in section IV. In section V we evaluate the implementation before we conclude in section VI.

II. Background and related work

In their taxonomy of DSM techniques [9] list four main approaches: (a) energy efficiency, (b) time of use, (c) demand

Support for this research was generously provided by Microsoft Azure

[1]The study's authors acknowledge that the worst-case scenario of 13% is 'exorbitant', but 'not totally unrealistic'

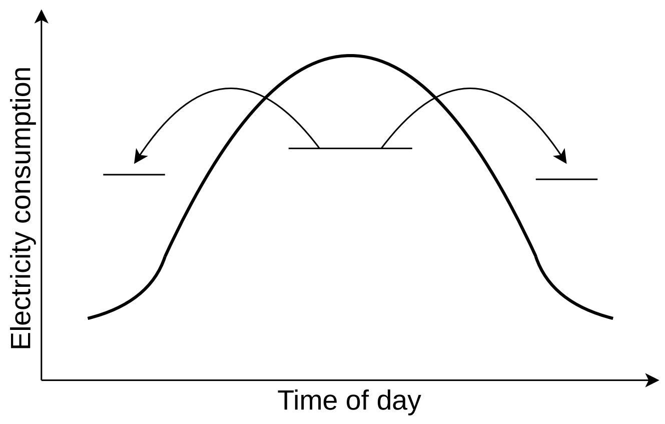

Fig. 1. Demand Side Managment (DSM) strategy - Load Shifting. The 'duck curve' of solar power generation can be observed, with energy generation peaking in the middle of the day [8]

response and (d) spinning reserve. Time of use refers to scheduling energy consumption outside of peak times. Demand response refers to reduction of electricity demand either via direct control of devices by the electric utility provider or using electricty tariffs in order to create incentives for consumers to alter their behaviour. Spinning reserve refers to the capability of some energy consuming devices to reduce their power consumption in response to changes to the grid frequency that results from load in the grid. Among these four categories, energy efficiency measures are most desirable as they result in long term reductions of energy consumption and thus cost.

A. Energy-efficient cloud computing

Greater datacentre energy efficiency may be achieved through a number of different methods. Some of these, compiled by Zakarya and Gillam [10], are outlined on Fig.2. Some of these methods will be further explored in relation to cloud computing in the proceeding section. The research outlined in this project chiefly pertains to the 'load balancing' and 'renewables' rows listed in Fig.2.

B. Green datacentres and renewable energy

Solar power generation is characterised by variability and uncertainty. Business decisions considering where best to install photovoltaic (PV) arrays rely on historical solar irradiation data, which measure the solar energy that reaches the earth's surface over a long-term period. This usable energy varies according to latitude, elevation, season, and climate. The value of more short-term, namely day-ahead, solar power forecasting is discussed in Brancucci et al.'s 2017 paper [11], and indicates that such forecasting can lead to a reduction in overall solar energy generation costs. The paper discusses the 'duck curve', in which solar power generation is observed to be highest during the middle of the day, and can account for a greater share of electrical power generation; however, more conventional power generation methods are required to meet demand during the downward (during sunrise) and upward (during sunset) sloping sections of the curve.

GreenSlot, proposed by Goiri et al. [12], is a scheduler which predicts the amount of solar energy that will be available in the near future, and schedules the workload to maximise green energy consumption while meeting the deadlines specified by the job submitter. Greenslot, however, operates within a datacentre rather than between distributed datacentres, as we propose. Additionally, the system was implemented for two specific schedulers (SLURM and the MapReduce scheduler of Hadoop); not for Kubernetes. GreenSlot increased green energy usage by 19-21% by delaying jobs so that they are executed during periods of high green energy production or low brown energy prices [13].

GreenWorks is a framework that was proposed by Li et al. [14] for datacentres powered by hybrid renewable energy systems. The framework considers the timing behaviours and capacity constraints of different energy sources that are available to a singular datacentre and makes optimal decisions based on the energy mix available at any time.

Wang et al. [15], consider a mix of green and brown energy sources, and use the following formula to implement a k-nearest neighbour (k-NN) based algorithm to forecast the solar energy level generation for the next day and make VM placement decisions accordingly.Additionally, their model is renewable-energy aware and considers the energy cost of datacentre cooling [15].

As highlighted by Brancucci et al. [11], Goiri et al. [13], Li et al. [14] and Wang et al. [15], several solar power forecasting technologies currently exist and are continuously improving, with modelling efforts accelerating thanks to advancements in machine learning techniques. Antonanzas et al. also consider very short term forecasting, denoted as intra-hour or 'nowcasting' [7]. All these works consider scheduling with regard to renewable energy sources, but do so with consideration to singular datacentres rather than taking a more global view, and deal with a mix of renewable and non-renewable energy sources rather than variable renewable energy sources exclusively [15].

The Low Carbon Scheduler on the other hand considers carbon intensity across regions as scaling up and down of a large number of containers can be done in a matter of seconds. Nonetheless, these works provide an understanding of green computing at the WSC level, knowledge of which can understand the trade-offs that are to be made in a geographically distributed scheduler.

C. Geographically distributed green datacentres

Reasoning that it is cheaper to transmit data over large distances than it is to transmit power, one of the first papers that suggested locating data centres near renewable energy sources was written in 2008 by Hopper and Rice [16]. As grids and datacentres are located in multiple regions that span the globe, each is powered by different mixes of both green and brown energy sources. Routing more user requests to the region

Fig. 2. Current approaches to datacentres energy efficiency [10]

Technique	*Explanation*	*Benefits*	*Shortcomings*
Virtualisation	Dynamically provision resources	Efficient energy saving	Widely used, VM live migration affects network performance
Server consolidation and encapsulating application	Reduces active servers by consolidating the workload of multiple servers	Increases the utilisation ratio of servers, reduce SLA violation ratio	Consequences from failure of single consolidated server
DCP (Dynamic Capacity Planning)	Adjust the available resources tocurrent demand	More energy-efficient	Involves cost of switching resources on/off; could violate customers' SLAs
Load Balancing	Balance the workload among different servers to level out average utilisation	Equal utilisation	Challenging to implement in a heterogeneous platform
Scheduling and VMs placement	Place VMs onto a suitable (most energy-efficient) servers	Server and communication system energy-efficient	Planning and live migration SLA violation
Live migrations	Migrate VMs from over-utilised & under-utilised to more efficient servers	Less energy consumption	Service level of running application affected
Renewables	Migrate VMs to servers operated by renewable energy sources	More energy efficient and economical	Renewables are intermittent & involve migrations that cost extra energy

which is powered by cheaper production technology not only helps to save energy, but, according to Zakarya and Gillam [10], offer at least three further benefits: (i) renewable energy in oversupply allows for energy to be fed back to the electricity grid; (ii) a high supply of renewables decreases demand for non-renewable sources from the electricity grid; (iii) lessened reliance on renewable energy storage reduces the costs of management and replenishment of storage mechanisms, such as batteries, and extends the life of these mechanisms.

Rahman et al. survey geographic load balancing of data centre workload [17]. Geographic load balancing for carbon reduction in the past has typically used request routing to direct demand relative to carbon intensity of electricity. Among them, [18] propose a traffic engineering framework, while [19] propose a conceptual model based on Simulated Annealing optimisation. Here, we go beyond the model and propose a solution on the level of the infrastructure orchestration provided by Kubernetes.

Berl et al. [20] outlines how a geographically-distributed workload allocation system could work, and proposed moving workload between datacentres if necessary in order to improve energy efficiency. The paper focuses not on scheduling in accordance with low carbon electricity, but advocates allocating work to cooler datacentres[2]. A 2014 Paper by Zhang, Shao et al. [22] estimated that 30 to 50% of a datacentre's energy consumption comprises 'cooling energy'. Oro and Salom came to a figure of 40% in 2015 [23]. For this reason the scheduler considers local air temperature when making placement decisions (see section IV-A).

'Green geographic load balancing' was also used in a paper by Islam et al. [24]; however, this was with the aim of rationing water consumption in datacentres rather than reducing carbon emissions. This is especially important during periods of drought, as experienced in California in recent years. Their algorithm, *WATCH* (WATer-constrained workload sCHeduling in data centers), dynamically dispatches workload across geographically distributed datacentres based on water availability. While this work does not directly relate to our proposed scheduler, the paper nonetheless demonstrates the feasibility of implementing a green scheduler, remembering that 'green' is a term that can be applied to minimising consumption of natural resources in addition to encouraging renewable energy usage.

In 2012, Van Heddeghem et al. concluded that while deploying additional datacentres can help in reducing total carbon footprint, substantial reductions could be achieved when datacentres with nominal capacity well below maximum capacity redistribute workload to sites based on the availability of renewable energy [25]. The authors take a probabilistic approach to chosing target data centres as opposed to our use of real-time API based on reported actual generation data.

EcoPower is a system designed to perform eco-aware power management and load scheduling for geographically distributed green cloud datacentres [26]. While the paper concludes that wind-dominant, solar-complementary strategy is superior for the integration of renewable energy sources into cloud datacentres' infrastructure, the Low Carbon Scheduler

[2]Facebook published some details of their datacentres in Sweden, which highlighted their efforts to locate their datacentres in colder climes in order to minimise datacentre cooling costs [21]

Country	Microsoft [29]	Google [30]	Amazon [31]	Oracle [32]
Australia	x	x		x
Belgium		x		
Brazil	x	x	x	x
Canada	x	x	x	
China	x		x	
Denmark		x		
Finland		x		
France	x		x	
Germany	x	x	x	x
India	x	x	x	
Ireland	x		x	
Japan	x		x	x
Korea	x		x	
Malaysia	x			
Netherlands		x		x
Norway	x			
Singapore		x		
South Africa	x			
Sweden			x	
Switzerland	x			
Taiwan	x			
UAE	x			
UK	x	x	x	x
US-CA	x	x	x	x
US-East	x	x	x	x
US-central	x	x	x	x

provides a proof-of-concept demonstrating *how* to reduce carbon intensity in cloud computing. Also, the Low Carbon Scheduler focuses on Kubernetes workloads, which is not the case with EcoPower. Calculating energy usage is also widely explored in the work of Khosravi et al. [27] on geographically distributed cloud datacentres.

Hasan et al. discuss green cloud computing from a business cloud user's perspective: companies may choose to specify a requirement for green energy usage in their Service Level Agreements (SLAs) with cloud computing providers [28]. In the paper they extend the Cloud Service Level Agreement (CSLA) language in order to incorporate two new threshold parameters that ensure that more environmentally sustainable policies are adhered to. The incentivisation of the Low Carbon Scheduler is discussed in section IV

D. *Geographically distributed cloud datacentres*

The largest public Cloud providers operate data centres around the planet. Table II-D lists the countries as of April 2019.

E. *Real-time carbon intensity*

Electricity in national electric grids is generated from a variable mix of alternative sources. The carbon intensity of the electricity provided by the grid anywhere in the world is a measure of the amount of greenhouse gas released into the atmosphere from the combustion of fossil fuels for the generation of electricty. The carbon intensity is calculated as the sum of the carbon intensity of the various energy sources weighted by the relative production volumes per energy source (i.e. fuel type). The dominant types of fossil fuel used for electricity generation are gas and coal. Significant generation sites (excluding, for example, domestic solar PV installations) report the volume of electricity input to the grid in regular intervals to the organisations operating the grid (for example the National Grid in the UK). Increasingly, this production data is made available in real-time via APIs. For the European Union such an API is provided by the European Network of Transmission System Operators for Electricity (www.entsoe.eu) and for the UK this is the Balancing Mechanism Reporting Service (www.elexon.co.uk). These APIs typically provide the retrieval of the production volumes and thus allow to calculate the carbon intensity in real-time [33]. Our low carbon scheduler collects the carbon intensity from the available APIs and ranks them to identify the region with lowest carbon intensity.

III. Design

Kubernetes has been adopted and adapted for the purpose of scheduling workload around the globe. While section III-C outlines the design decisions made in order to enable a low carbon scheduling policy, a brief overview of Kubernetes and the role of scheduling within Kubernetes is provided first.

A. *Kubernetes and container orchestration*

Kubernetes, initially developed by Google and open-sourced in 2015, is based on the company's experience of running containers internally on Google's own WSCs using its proprietary Borg system [34]. The source code for Kubernetes was released in July 2015, and has grown to have more pull requests and issue comments than any of the 54 million other projects on GitHub [35]. Kubernetes was later donated to the Cloud Native Computing Foundation, part of the Linux Foundation.

The user provides the Kubernetes master[3] with the desired cluster configuration, typically in YAML format. Once the desired state has been declared to the master, Kubernetes initiates a reconciliation process to match the desired state of the cluster with the current, actual, state. Once the desired cluster state has been achieved, the Kubernetes controllers are in an active reconciliation process, i.e. they monitor for changes made to either the desired state (through user input) or the current state (through node or netwok failures, for example), and ensures that if a change is detected, the Kubernetes controllers carries out the required operations to match the cluster's current state with its desired state [36].

Kubernetes can make use of GPUs[4] and has also been ported to run on ARM architecture[5]. Kubernetes has to a large extent won the container orchestration war [41], [42]. This, coupled with Kubernetes's support for extendability and plugins makes Kubernetes the most suitable for which to develop a global scheduler and bring about the widest adoption, thereby producing the greatest impact on carbon emission reduction.

[3]Through CLI/GUI/API

[4]Used extensively for Machine Learning and other GPU-intensive tasks such as graphics rendering - NVIDIA have released a container for use with GPUs [37]. Microsoft has made use of Kubernetes for running deep learning models [38]. Kubernetes has also been used with great success for bioinformatic analysis [39]

[5]Which uses substantially less power than the CPUs of traditional server and desktop computers [40]

Term	Definition
Pod	The atomic unit of a Kubernetes cluster; a group of one or more containers with shared storage/network, and a specification for how to run the containers nested within the pod
Master	Provides the cluster's control plane. Master components make global decisions about the cluster (for example, scheduling), and detecting and responding to cluster events, such as restarting stopped pods.

Fig. 3. Kubernetes architecture and basic terminology [43]

As outlined on Fig. 3, the Kubernetes master performs a number of roles, among them scheduling. Kubernetes allows for schedulers to run in parallel, meaning that the scheduler will not need to re-implement the pre-existing, and sophisticated, bin-packing strategies present in Kubernetes. It need only apply a scheduling layer to compliment the existing capabilities proffered by Kubernetes.

B. Scheduling in Kubernetes

Kubernetes builds on work that was done at Google for managing its internal cluster, called Borg [34], and later on a project called Omega[6] [45]. Facebook is believed to use a similar service called Tupperware [34]. Google's publication of 'Large-scale Cluster Management at Google with Borg' [34] proved to be seminal, and is counted as the key publication on which Kubernetes is based. A number of features and concepts from Borg have been brought forward to Kubernetes, including API Servers, Pods, IP-per-Pod, Services, Labels [34]. The Omega paper also provides a useful description of scheduler interference [45], whereby multiple schedulers may attempt to claim the same resource simultaneously. The Omega paper explains that two approaches can be used to mitigate this: a pessimistic approach which ensures that a particular resource is only made available to one scheduler at a time, and an optimistic approach, which detects conflicts and undoes one or more of the conflicting claims [45]. Our design, as it operates at a higher level of abstraction, assures that Kubernetes continues to deal with bin-packing at the node level, while the scheduler performs global-level scheduling between datacentres.

The default scheduling algorithm used by Kubernetes is succinctly explained in a README file in the source code:

> *There are two steps before a destination node of a Pod is chosen. The first step is filtering all the nodes and the second is ranking the remaining nodes to find a best fit for the Pod.* [46]

The scheduler evaluates all the nodes in the cluster based on a number of rules, known as *Predicates*; these are scheduling rules that filter out unqualified nodes[7]. Another set of scheduling rules are called *Priorities*; these are scheduling rules that rank the remaining nodes according to preferences[8]. A scheduling policy is a particular combination of predicates and priorities.

The scheduler specifically

- (a) looks for Pods that aren't assigned to a node (unbound Pods),
- (b) examines the state of the cluster (cached in memory),
- (c) picks a node that has free space and meets other constraints
- (d) binds that Pod to a node. If multiple nodes are assigned the same priority, a node is chosen at random [47]

C. Extending the Kubernetes scheduler

The official Kubernetes documentation describes three possible ways of extending the default scheduler (kube-scheduler) [48]: (i) adding these rules to the scheduler source code and recompiling, (ii) implementing one's own scheduler process that runs instead of, or alongside kube-scheduler, or (iii) implementing a scheduler extender.

D. Air temperature and solar irradiance

As described in the literature review in section II-C, the local air temperature surrounding a datacentre affects the amount of energy needed for cooling; air temperatureis therefore a relevant consideration when the scheduler selects the most suitable datacentre for workload allocation. In the scheduler's design, two datacentres with similarly-carbon intense grid electricity are further ranked by temperature, with the cooler location prioritised for the (re)allocation of the specified workload.

E. Carbon emission model

In this subsection we describe a brief conceptual model of the carbon emissions associated to computation and migration of work.

Carbon emissions that result from the consumption of electric energy can be calculated as the product of the electric energy E and the carbon intensity of electricity I, thus $E \cdot I$. Compute work drives the consumption of electric energy EC (*energy compute* in data centres and networks mainly in three ways. Most importantly, electric energy is required for servers during the runtime t of the computation. The power consumption P is a function of the varying utilisation of compute resources (e.g. CPU, memory, IO) over time. Thus, $EC = \int P(u(t))dt$. Secondly, electric energy is also consumed during transmission over the network of any data (input to or results of computation), labeled EN (*energy network*). This energy consumption is proportional to the volume of data transferred [49]. Finally, there is a ramp-up overhead from deploying the Kubernetes service in the target location ER.

[6]Kubernetes is in fact claimed to be in many ways superior to Borg and Omega [44]

[7]PodFitsResources, PodFitsPorts, MatchNodeSelector etc.

[8]Among the most commonly used are ImageLocalityPriority, BalancedResourceAllocation, LeastRequestedPriority

Carbon emissions can be reduced by migrating a Kubernetes deployment if

$$EC_A I_A > EC_B I_B + ER_B + EN_{AB} I_{AB}$$

with EC_A, EC_B the compute energy in data centre A and B, I_A, I_B the carbon intensities in regions of data centre A and B, ER_B the energy consumed for deploying the Kubernetes service, EN_{AB} the energy consumed for transporting all required data from A to B and I_{AB} the average carbon intensity in the network route from A to B [50].

In this model we assume PUE is similar between data centre A and B. Among these variables, the carbon intensities are known to a Cloud *customer* via the carbon intensity APIs. Cloud providers, i.e. data centre operators will also be able to determine any differences between EC_A and EC_B, ER_B. Cloud operators would also know if PUE (power utilisation efficiency) differs between two locations. PUE factors can simply be added as coefficients to either side of the relation.

In our evaluation we present a concrete service for which very little data has to be transported during migration of the Kubernet deployment, and that thus can be optimised from the perspective of the Cloud customer, in other words, based on knowledge of I_A and I_B alone, assuming that the runtime energy consumption for identical computation in two locations A and B is similar, thus $EC_A \approx EC_B$.

F. Pseudo-code

Algorithm 1 The Low Carbon Kubernetes Scheduler

```
Require: $kubectl$
Require: $cloudproviderCLI$
  $P = (x, y)$
  $I_D$
  $greenestregion =$
  for all $P$ do
    $get\_carbon\_intensity$
    for all $P$ do
      if $I_D = 0$ then
        delete
      end if
      $sort\_by\_carbon\_intensity$
      if $I[loc0] \approxeq I[loc0]$ then
        for all $P$ do
          $sortbyairtemp$
        end for
        return $topregion$
      else
        return $topregion$
      end if
    end for
  end for
  $wait30mins$
```

IV. Implementation

In this section we describe the current implementation of the design. At present the implementation is compatible with the APIs of the Azure platform. The introduction of additional public clouds is straight forward as described in IV-B.

1

The scheduler receives the carbon intensity values for all viable datacentre regions. Once the results have been received, the scheduler ranks the locations. By default this ranking occurs in accordance carbon intensity and air temperature, but can be modified, as demonstrated later.

Having determined the most suitable (i.e. 'greenest') datacentre location, the program sends a request to the cloud Kubernetes or IaaS management API to provision a Resource Group at that datacentre, then verifies that this was successful. Upon confirming the success of Resource Group creation a request to provision a Kubernetes cluster is sent. Typically, a new cluster takes around 10 minutes to provision and for the credentials to be agreed upon[9], and often an additional minute or two for all of Kubernetes's internal components to be in a 'Ready' status. In order to wait for this to happen, the scheduler polls the cluster at regular intervals[10] for the status of its components. Once the cluster is in 'Ready' state the specified Deployment is executed. After all resources have been created the Scheduler deletes the Resource Group in the region that was just determined to be less suitable[11]. This design ensures that the next cluster is fully up and deployed before pulling down the previous cluster, ensuring that the deployment is running continuously. It also addresses the issue of what would happen if such a scheduler were widely used, and if a large number of users were demanding resources from the same datacentre: if the datacentre were overburdened with requests, it would simply return a message to indicate that the deployment cannot be placed, allowing the workload to continue as normal in the previous region.

AKS (Azure Container Service) [sic] - Managed Kubernetes.. AKS reduces the complexity and operational overhead of managing a Kubernetes cluster; however, this is only currently offered in three regions [12]. Also unable to scale down to zero pods[13]

A. Incentivisation

In the first instance, the Low Carbon Scheduler will be appealing organisations aiming to increase the sustainability of their compute jobs. It could also play a role in ensuring that such green SLAs are adhered to by allowing some companies to opt in to a greener scheduling policy. One such proposal for the Low Carbon Kubernetes Scheduler could be allowing the deployer of cloud resources to declare their deployment as 'latency-insensitive', which would permit cloud operators to

[9]Using the host's public SSH key, or a user-specified public SSH key

[10]in our implementation, every 20 seconds

[11]Deleting the cluster alone won't necessarily remove the cluster's child resources. Ideally, in all areas that cannot generate renewable energy, CPU cycles of any kind (including those made by the Kubernetes master) would be zero. This would be unnecessarily costly and energy-inefficient.

[12]As of March 2018: centralus, eastus, westeurope [51]

[13]CLI says it must be 1 or greater [52]

schedule that workload in a manner that optimises demand-side decarbonisation[14] .

B. Extensibility

The software has been written to allow for easy extensibility. Further metrics can be introduced to the code in order to influence the datacentre scheduling decisions. The software's *plugin* package contains variables and suggested function declarations that would allow practically any kind of metric to be passed to it, similar to the way that the Kubernetes scheduler does. It would be possible, for example, to introduce consideration of live cloud-region pricing data posted on AzurePrice.net. Extensibility of the scheduler is important in order to allow new metrics to be introduced to influence scheduling decisions. Some metrics, for example, are simply not available to the public[15], but would be useful for the implementation of a carbon-aware scheduling policy [53].

In order to facilitate extensibility of cloud providers beyond Azure, the source code strives to ensure that vendor-specific commands are kept to their own packages. Azure-specific commands are contained in the *azacs*[16] package. Other developers may then easily add functionality to the scheduler by introducing new packages for each cloud vendor. Additionally, once AKS[17] is supported in a greater number of regions, it would be a trivial task to customise the source code to use AKS instead of (or in addition to) ACS.

It would have been possible to configure the scheduler to pull the deployment specification YAML from the running cluster, and pass this configuration onto the next region, but storing the file in a GitHub 'gist' that the scheduler is aware of makes use of the practice in cloud computing known as 'infrastructure as code'. This relates to managing and provisioning resources using definition files, rather than physical hardware configuration. This makes builds more reproducable and allows for version control systems to be used to track files and reverse a non-functioning declaration to a previous, working state. The scheduler can be easily configured to specify either a URL pointing to a raw YAML or JSON file, or to specify a locally-stored deployment configuration.

V. Evaluation

A. Carbon Ranks

We recorded the carbon intensities for the countries that the major cloud providers operate data centres in (see II-D) between 18.2.2019 13:00 UTC and 21.4.2019 9:00 UTC. We then ranked all countries by the carbon intensity of their electricity in 30 minute intervals. Among the total set of 30 minute values Switzerland had the lowest carbon intensity (ranked first) in 0.57% of the 30 minute intervals, Norway 0.31%, France 0.11% and Sweden in 0.01%.

[14]Conceivably at a fractionally lower cost in order to incentivise its usage

[15]Such as each datacentre's green/brown energy mix and how much energy storage capacity is at each location

[16]Azure ACS (Azure Container Service)

[17]Azure's managed Kubernetes service

B. The Heliotropic Scheduler

The list of least carbon intense countries only contains countries in central Europe locations. In our evaluation of the Kubernetes extension and its ability for globally distributing deployments we have chosen to optimise placement to regions with the greatest degree of solar irradiance, termed a Heliotropic Scheduler. Solar irradiance varies more widely than carbon intensity across global regions.

This scheduler is termed 'heliotropic' in order to differentiate it from a 'follow-the-sun' application management policy as mentioned in the documentation to the cloud framework Apache Brooklyn [54] [55] and in academic work [56]. While 'follow-the-sun' relates to meeting customer demand around the world by placing staff and resources in proximity to those locations (thereby making them available to clients at a lower latency and at a suitable time of day), a 'heliotropic' policy goes to where sunlight, and by extension solar irradiance, is abundant.

1) Live solar irradiance data: As the scheduler reacts to changes in insolation in near real time, a good source of live weather data is crucial for its correct functioning. Following a review of seven live weather APIs [57], Weatherbit.io was chosen as it was the sole simulatenous provider of three metrics necessary for the Heliotropic Scheduler: air temperature, windspeed, and live insolation data. This latter, crucial measurement was derived from a metric called DHI, or 'Diffuse Horizontal Irradiance'. DHI signifies the amount of radiation received on a horizontal surface that does not arrive on a direct path from the sun, but has been scattered by molecules and particles in the atmosphere [58]; it roughly corresponds to Watts generated per square metre [18] [59]. The veracity of the insolation data provided by Weatherbit.io could be verified by comparison with equivalent data from other Weather API providers.

C. BOINC

We evaluate our implementation of the Heliotropic Scheduler by running BOINC[19] jobs on Kubernetes. BOINC (Berkeley Open Infrastructure for Network Computing) is a software platform for volunteer computing that allows users to contribute computational capacity from their home PCs (usually when the computer is idle) towards scientific research [60]. Among the most widely supported projects are Einstein@Home, SETI@home and IBM World Community Grid[20].

While any number of programs could have been chosen or written to carry out compute workload on the heliotropically-scheduled cluster, BOINC was chosen, along with the IBM Community Grid project, so that the project might contribute to scientific research rather than perform an arbitrary 'number-crunching' task of our own design. The BOINC client down-

[18]Subsequent investigations into the Weatherbit API revealed that additional solar insolation metrics (DNI and GHI) were provided, but undocumented on the Weatherbit website

[19]rhymes with 'oink'

[20]As of January 2, 2018, 37 BOINC projects are active [61]

loads raw data, processes them and then uploads the results back to the project servers before requesting additional work [62]. Choosing BOINC as the cluster workload therefore offers the advantage of there being no strong requirement for either low latency or persistent storage.

A paper for further research regarding volunteer computing (specifically BOINC) in the cloud, by Montes, Añel et al. [63], demonstrates the suitability of BOINC for cloud computing in certain circumstances[21]. This project's work on BOINC [64], including a Dockerfile and publically available image, are available on Docker Hub[22].

D. *Results of evaluatory experiments*

Pages 8 to 9 show empirical results of the Heliotropic Scheduler placing workload in Microsoft Azure datacentres across the globe. Each column of graphs shows the varying DHI, the deployment location for the BOINC cluster over time together with a map of the datacentre locations. The first two tests show how the scheduler correctly identified the most suitable region based on insolation and allocated work to those regions as desired in the design specifications. Fig. 4 shows that the deployment was raised in australiaeast, in accordance with DHI, and remained there for the duration of the test.

Fig. 5 shows that the deployment was raised in westeurope, in accordance with DHI, before scheduling itself heliotropically to eastus, and later centralus.

Fig. 6 demonstrates the scheduler's extensibility. With a minimal amount of configuration, the scheduler operated on a follow-the-wind model. As wind power continues to be generated at night, a greater number of datacentres are in contention to be the most suitable. For this reason a number of redeployments occur over the test's time period. Depending on the nature of the work, datacentre migration might include the transfer of a significant amount of data. In this case, reallocation thresholds can limit the number of migrations that can occur over a period of time.

VI. Conclusion

We presented the design and implementation of a low carbon scheduling policy for the open-source Kubernetes container orchestrator. The implementation is fully functional and could successfully migrate a Kubernetes deployment between global regions.

For cloud customers, the current optimisation model of the scheduler is robust of for workloads that do not require significant data transport as part of the migration - such as is the case with the BOINC workloads. Even though many cloud providers are contracting for renewable energy with their energy providers, the electricity these data centres take from the grid is generated with release of a varying amount of greenhouse gas emissions into the atmosphere. Our scheduler can contribute to moving demand for more carbon intense electricity to less carbon intense electricity.

[21]The paper looks at running the BOINC client on Amazon's Cloud Services platform (AWS), and contributes to the ClimatePrediction.net project

[22]Docker Hub is a centralised resource for public and private container images

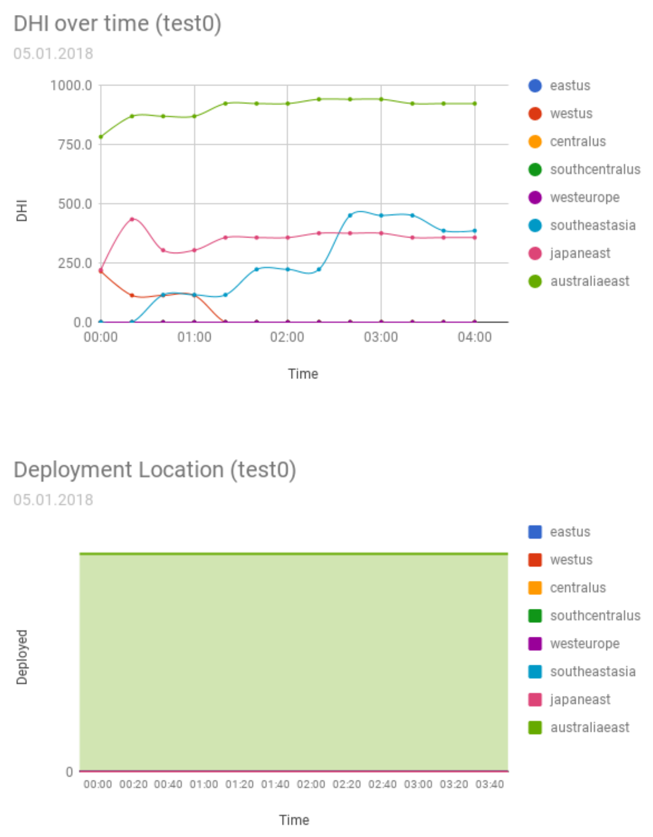

Fig. 4. Test 0 (DHI)

References

[1] L. A. Barroso, J. Clidaras, and U. Hölzle, *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines, Second Edition*. Morgan and Claypool, 2013. [Online]. Available: http://dx.doi.org/10.2200/S00516ED2V01Y201306CAC024

[2] A. S. G. Andrae and T. Edler, "On global electricity usage of communication technology: Trends to 2030," *Challenges*, vol. 6, no. 1, pp. 117–157, 2015. [Online]. Available: http://www.mdpi.com/2078-1547/6/1/117

[3] Urs Hölzle, "We're set to reach 100 percent renewable energy — and it's just the beginning," 2016. [Online]. Available: https://www.blog.google/topics/environment/100-percent-renewable-energy/

[4] Microsoft, "Addressing our carbon footprint," 2017. [Online]. Available: https://www.microsoft.com/about/csr/environment/carbon/

[5] Oracle, "Move to the Cloud for Energy Efficiency," 2017. [Online]. Available: https://www.oracle.com/solutions/green/cloud-operations.html

[6] A. S. G. Andrae and T. Edler, "Google environment report: 2017 progress update," 2017. [Online]. Available: https://static.googleusercontent.com/media/environment.google/en//pdf/google-2017-environmental-report.pdf

[7] J. Antonanzas and N. Osorio and R. Escobar and R. Urraca and F.J. Martinez-de-Pison and F. Antonanzas-Torres, "Review of photovoltaic power forecasting," *Solar Energy*, vol. 136, pp. 78 – 111, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0038092X1630250X

[8] World Bank, "Primer on Demand-Side Management: With an Emphasis on Price-Responsive Programs," *World Bank Other Operational Studies*, 2005. [Online]. Available: http://documents.worldbank.org/curated/en/986041468154163610/

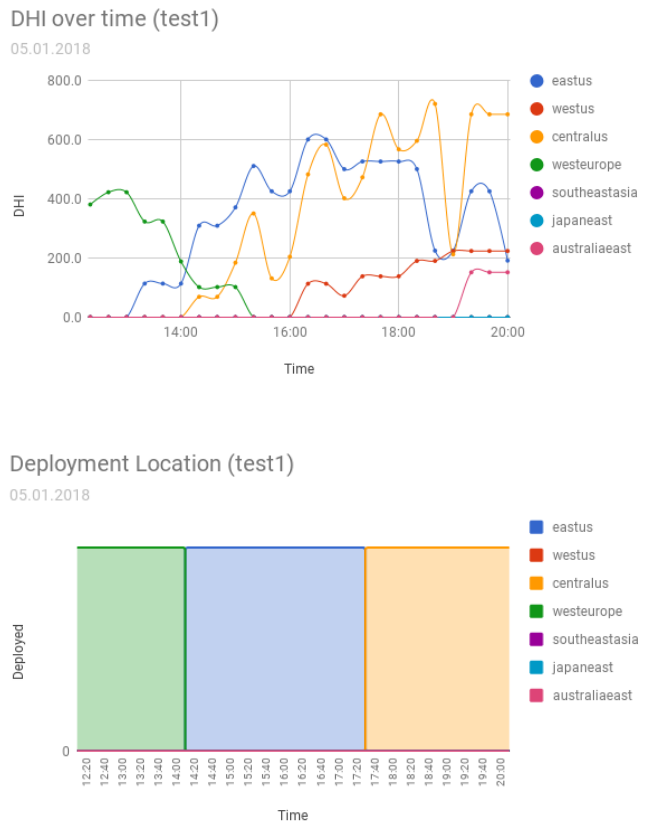

Fig. 5. Test 1 (DHI)

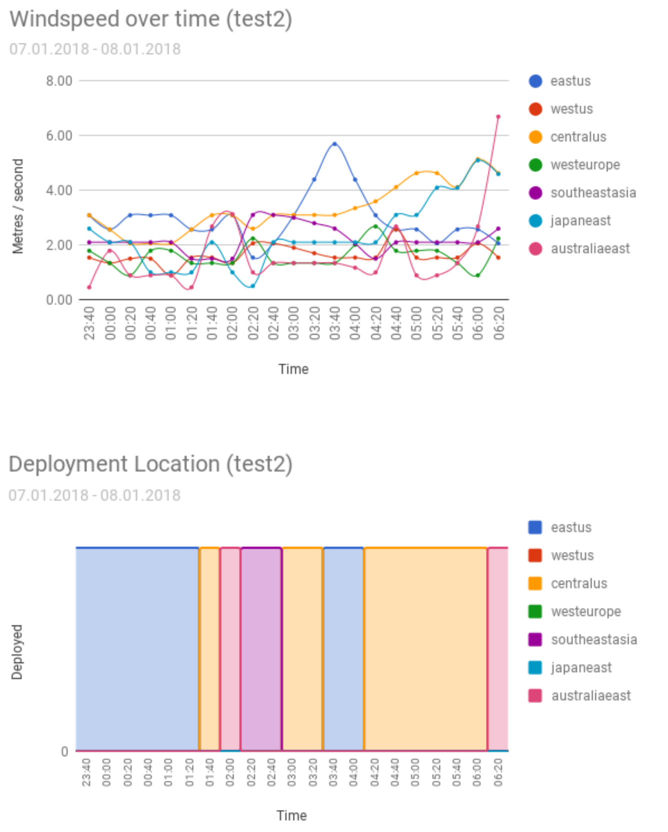

Fig. 6. Test 2 (DHI)

Primer-on-demand-side-management-with-an-emphasis-on\protect\discretionary{\char\hyphenchar\font}{}{}price-responsive-programs

[9] P. Palensky and D. Dietrich, "Demand side management: Demand response, intelligent energy systems, and smart loads," *IEEE Transactions on Industrial Informatics*, vol. 7, no. 3, pp. 381–388, 2011.

[10] M. Zakarya and L. Gillam, "Energy efficient computing, clusters, grids and clouds: A taxonomy and survey," *Sustainable Computing: Informatics and Systems*, vol. 14, pp. 13 – 33, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2210537917300707

[11] Carlo Brancucci Martinez-Anido and Benjamin Botor and Anthony R. Florita and Caroline Draxl and Siyuan Lu and Hendrik F. Hamann and Bri-Mathias Hodge, "The value of day-ahead solar power forecasting improvement," *Solar Energy*, vol. 129, pp. 192 – 203, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0038092X16000736

[12] I. Goiri, R. Beauchea, K. Le, T. D. Nguyen, M. E. Haque, J. Guitart, J. Torres, and R. Bianchini, "Greenslot: Scheduling energy consumption in green datacenters," in *2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, Nov 2011, pp. 1–11.

[13] I. Goiri, M. E. Haque, K. Le, R. Beauchea, T. D. Nguyen, J. Guitart, J. Torres, and R. Bianchini, "Matching renewable energy supply and demand in green datacenters," *Ad Hoc Networks*, vol. 25, pp. 520 – 534, 2015, new Research Challenges in Mobile, Opportunistic and Delay-Tolerant Networks Energy-Aware Data Centers: Architecture, Infrastructure, and Communication. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870514002649

[14] C. Li, R. Wang, D. Qian, and T. Li, "Managing server clusters on renewable energy mix," *ACM Trans. Auton. Adapt. Syst.*, vol. 11, no. 1, pp. 1:1–1:24, Feb. 2016. [Online]. Available: http://doi.acm.org/10.1145/2845085

[15] X. Wang, Z. Du, Y. Chen, and M. Yang, "A green-aware virtual machine migration strategy for sustainable datacenter powered by renewable energy," *Simulation Modelling Practice and Theory*, vol. 58, pp. 3 – 14, 2015, special Issue on Techniques and Applications for Sustainable Ultrascale Computing Systems. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1569190X15000155

[16] A. Hopper and A. Rice, "Computing for the future of the planet," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 366, no. 1881, pp. 3685–3697, 2008. [Online]. Available: http://rsta.royalsocietypublishing.org/content/366/1881/3685

[17] A. Rahman, X. Liu, and F. Kong, "A survey on geographic load balancing based data center power management in the smart grid environment," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 1, pp. 214–233, 2014.

[18] S. K. Peter Xiang Gao, Andrew R. Curtis, Bernard Wong, "It's Not Easy Being Green," in *SIGCOMM*, 2012, pp. 2011–2013.

[19] K. Le, R. Bianchini, T. D. Nguyen, O. Bilgir, and M. Martonosi, "Capping the brown energy consumption of internet services at low cost," *2010 International Conference on Green Computing, Green Comp 2010*, pp. 3–14, 2010.

[20] A. Berl, E. Gelenbe, M. Di Girolamo, G. Giuliani, H. De Meer, M. Q. Dang, and K. Pentikousis, "Energy-efficient cloud computing," *The Computer Journal*, vol. 53, no. 7, pp. 1045–1051, 2010. [Online]. Available: http://dx.doi.org/10.1093/comjnl/bxp080

[21] Mark Zuckerberg, "Luleå data center." September 2016. [Online]. Available: https://www.facebook.com/zuck/posts/10103136694875121

[22] H. Zhang, S. Shao, H. Xu, H. Zou, and C. Tian, "Free cooling of data centers: A review," *Renewable and Sustainable Energy Reviews*, vol. 35, pp. 171 – 182, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1364032114002445

[23] E. Oró and J. Salom, "Energy model for thermal energy storage system management integration in data centres," *Energy Procedia*, vol. 73, pp. 254 – 262, 2015, 9th International Renewable Energy Storage Conference, IRES 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1876610215014526

[24] M. A. Islam, S. Ren, G. Quan, M. Z. Shakir, and A. V. Vasilakos, "Water-constrained geographic load balancing in data centers," *IEEE Transactions on Cloud Computing*, vol. 5, no. 2, pp. 208–220, April 2017.

[25] W. V. Heddeghem, W. Vereecken, D. Colle, M. Pickavet, and P. Demeester, "Distributed computing for carbon footprint reduction by exploiting low-footprint energy availability," *Future Generation Computer Systems*, vol. 28, no. 2, pp. 405 – 414, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X11000859

[26] X. Deng, D. Wu, J. Shen, and J. He, "Eco-aware online power management and load scheduling for green cloud datacenters," *IEEE Systems Journal*, vol. 10, no. 1, pp. 78–87, March 2016.

[27] A. Khosravi, L. L. H. Andrew, and R. Buyya, "Dynamic vm placement method for minimizing energy and carbon cost in geographically distributed cloud data centers," *IEEE Transactions on Sustainable Computing*, vol. 2, no. 2, pp. 183–196, April 2017.

[28] Md Sabbir Hasan, and Yousri Kouki, and Thomas Ledoux, and Jean-Louis Pazat, and undefined, and undefined, and undefined, and undefined, , "Exploiting Renewable Sources: When Green SLA Becomes a Possible Reality in Cloud Computing," *IEEE Transactions on Cloud Computing*, vol. 5, no. 2, pp. 249–262, 2017.

[29] Microsoft, "Microsoft Azure Data Centre Locations," 2019. [Online]. Available: https://azure.microsoft.com/en-gb/global-infrastructure/regions/

[30] Google, "Google Cloud Data Centre Regions," 2019. [Online]. Available: https://cloud.google.com/about/locations/

[31] Amazon Web Services, "Amazon Web Service Data Centre Locations," 2019. [Online]. Available: https://aws.amazon.com/about-aws/global-infrastructure/

[32] Oracle, "Oracle Cloud Data Centre Locations," 2019. [Online]. Available: https://cloud.oracle.com/data-regions

[33] B. Tranberg, O. Corradi, B. Lajoie, T. Gibon, I. Staffell, and G. B. Andresen, "Real-Time Carbon Accounting Method for the European Electricity Markets," 2018. [Online]. Available: http://arxiv.org/abs/1812.06679

[34] Abhishek Verma, Luis Pedrosa, Madhukar Koupolu et al., "Large-scale Cluster Management at Google with Borg," 2015. [Online]. Available: http://research.google.com/pubs/archive/44843.pdf

[35] Sarah Novotny, "Happy Second Birthday: A Kubernetes Retrospective," 2017. [Online]. Available: http://blog.kubernetes.io/2017/07/happy-second-birthday-kubernetes.html

[36] The Kubernetes Authors, "Writing Controllers," 2017. [Online]. Available: https://github.com/kubernetes/community/blob/8decfe4/contributors/devel/controllers.md

[37] NVIDIA and Open Source Contributors, "Build and run Docker containers leveraging NVIDIA GPUs," 2017. [Online]. Available: https://github.com/NVIDIA/nvidia-container-runtime

[38] William Buchwalter and Rita Zhang, "Autoscaling Deep Learning Training with Kubernetes," November 2017. [Online]. Available: https://www.microsoft.com/developerblog/2017/11/21/autoscaling-deep-learning-training-kubernetes/

[39] O. Markstedt, J. Persson, J. Andersson, and O. Spjuth, "Kubernetes as an approach for solving bioinformatic problems." Uppsala universitet, Teknisk-naturvetenskapliga vetenskapsområdet, Biologiska sektionen, Institutionen för biologisk grundutbildning, 2017.

[40] Lucas Käldström, "Kubernetes on ARM Project," 2017. [Online]. Available: https://github.com/luxas/kubernetes-on-arm

[41] Portworx, "2017 Annual Container Adoption Survey: Huge Growth in Containers," April 2017. [Online]. Available: https://portworx.com/2017-container-adoption-survey/

[42] T. Krazit, "The Cloud in 2017: Amazon Web Services shows no signs of slowing during the year of Kubernetes," December 2017. [Online]. Available: https://www.geekwire.com/2017/cloud-2017-amazon-web-services-shows-no-signs-slowing-year-kubernetes/

[43] The Kubernetes Authors, "Kubernetes Documentation (Home)," 2017. [Online]. Available: https://kubernetes.io/docs/home/

[44] E. Brewer, "Google systems guru explains why containers are the future of computing," 2015. [Online]. Available: https://medium.com/s-c-a-l-e/google-systems-guru-explains-why-containers-are-the-future-of\-computing-87922af2cf95

[45] M. Schwarzkopf, A. Konwinski, M. Abd-El-Malek, and J. Wilkes, "Omega: flexible, scalable schedulers for large compute clusters," in *SIGOPS European Conference on Computer Systems (EuroSys)*, Prague, Czech Republic, 2013, pp. 351–364. [Online]. Available: http://eurosys2013.tudos.org/wp-content/uploads/2013/paper/Schwarzkopf.pdf

[46] The Kubernetes Authors, "Scheduler Algorithm in Kubernetes," 2017. [Online]. Available: https://github.com/kubernetes/community/blob/master/contributors/devel/scheduler_algorithm.md

[47] Joe Beda, "Core Kubernetes: Jazz Improv over Orchestration," 2017. [Online]. Available: https://blog.heptio.com/core-kubernetes-jazz-improv-over-orchestration-a7903ea92ca

[48] The Kubernetes Authors, "Scheduler extender," 2017. [Online]. Available: https://github.com/kubernetes/community/blob/master/contributors/design-proposals/scheduling/scheduler_extender.md

[49] D. Schien and C. Preist, "Approaches to energy intensity of the internet," *IEEE Communications Magazine*, vol. 52, no. 11, pp. 130–137, nov 2014. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6957153

[50] D. Schien, C. Preist, M. Yearworth, and P. Shabajee, "Impact of Location on the Energy Footprint of Digital Media," in *IEEE International Symposium on Sustainable Systems and Technology (IEEE ISSST 2012)*. Boston, MA: IEEE, 2012.

[51] Microsoft, "Azure status," January 2018. [Online]. Available: https://azure.microsoft.com/en-gb/status/

[52] Microsoft Azure and Open Source Contributors, "'az aks scale' should be able to scale #of linux and windows nodes #79," 2017. [Online]. Available: https://github.com/Azure/AKS/issues/79

[53] C. Bussar, M. Moos, R. Alvarez, P. Wolf, T. Thien, H. Chen, Z. Cai, M. Leuthold, D. U. Sauer, and A. Moser, "Optimal allocation and capacity of energy storage systems in a future european power system with 100% renewable energy generation," *Energy Procedia*, vol. 46, pp. 40–47, 2014. [Online]. Available: http://www.sciencedirect.com/journal/energy-procedia/vol/46

[54] Apache, "Apache Brooklyn Source Code: followthesun directory," 2017. [Online]. Available: https://github.com/apache/brooklyn-server/tree/master/policy/src/main/java/org/apache/brooklyn/policy/followthesun

[55] ——, "The Theory behind Brooklyn," 2017. [Online]. Available: https://brooklyn.apache.org/learnmore/theory.html

[56] E. Carmel, J. A. Espinosa, and Y. Dubinsky, ""Follow the Sun" Workflow in Global Software Development," *Journal of Management Information Systems*, vol. 27, no. 1, pp. 17–38, 2010. [Online]. Available: http://www.tandfonline.com/doi/abs/10.2753/MIS0742-1222270102

[57] Todd Motto and Open Source Contributors, "A collective list of public JSON APIs for use in web development," 2017. [Online]. Available: https://github.com/toddmotto/public-apis#weather

[58] Bill Marion, Carol Riordan and David Renne, "Shining On: A Primer on Solar Radiation Data," 1992. [Online]. Available: https://www.nrel.gov/docs/legosti/old/4856.pdf

[59] Weatherbit.io, "Current Weather API," 2017. [Online]. Available: https://www.weatherbit.io/api/weather-current

[60] U. of California, "Boinc: Open-source software for volunteer computing," 2017. [Online]. Available: http://boinc.berkeley.edu/

[61] ——, "Choosing boinc projects," 2018. [Online]. Available: http://boinc.berkeley.edu/projects.php

[62] ——, "How boinc works," 2017. [Online]. Available: http://boinc.berkeley.edu/wiki/How_BOINC_works

[63] D. Montes, J. A. Añel, T. F. Pena, P. Uhe, and D. C. H. Wallom, "Enabling boinc in infrastructure as a service cloud system," *Geoscientific Model Development*, vol. 10, no. 2, pp. 811–826, 2017. [Online]. Available: https://www.geosci-model-dev.net/10/811/2017/

[64] obfuscated, "Docker hub public repository: obfuscated," 2018. [Online]. Available: https://hub.docker.com/r/obfuscated/