Technical Report

UCAM-CL-TR-889
ISSN 1476-2986

Number 889

Evaluating the viability of remote renewable energy in datacentre computing

Sherif Akoush, Ripduman Sohan,
Andrew Rice, Andy Hopper

May 2016

ISSN 1476-2986

Evaluating the Viability of Remote Renewable Energy in Datacentre Computing

Sherif Akoush, Ripduman Sohan, Andrew Rice and Andy Hopper

Abstract

We investigate the feasibility of loosely-coupled distributed datacentre architectures colocated with and powered by renewable energy sources and interconnected using high-bandwidth low-latency data links. The design of these architectures advocates (*i*) variable machine availability: the number of machines accessible at a particular site at any given time is proportional to the amount of renewable energy available and (*ii*) workload deferment and migration: if there is insufficient site capacity, workloads are either halted or shifted to transient energy-rich locations.

While these architectures are attractive from an environmental perspective, their feasibility depends on (*i*) the requirement for additional hardware, (*ii*) the associated service interruptions, and (*iii*) the data and energy overhead of workload migration.

In this work we attempt to broadly quantify these overheads. We define a model of the basic design of the architecture incorporating the energy consumption of machines in the datacentre and the network. We further correlate this energy consumption with renewable energy available at different locations around the globe. Given this model we present two simulation-driven case studies based on data from Google and Facebook production clusters.

Generally we provide insights on the trade-offs associated with this off-grid architecture. For example, we show that an optimised configuration consisting of ten distributed datacentres results in a 2% increase in job completion time at the cost of a 50% increase in the number of machines required.

1 Introduction

Datacentres accounted for 1.5% of global electricity consumption in 2011; 56% higher than the previous five years [1]. Energy efficiency gains are not matching the increasing demand for computing [2]; while technologies such as smart standby and dynamic frequency [3, 4] scaling promise at best to provide energy-proportional computing. Consequently, datacentre operators incur high energy bills and face environmental pressure from society.

Researchers have thus started exploring the use of renewable energy in datacentres as an alternative clean and potentially cheap energy source [5, 6, 7, 8, 9, 10, 11]. However intermittent renewable energy adds hardship to the "always on" guarantees by datacentres. This challenge motivated a plethora of research directions to find a viable solution.

The first track of research focuses on smart scheduling of the computing load. As many workloads are delay tolerant, jobs can effectively execute when energy is available [10, 7, 11]. However the fact that renewable energy is not predictable can cause large periods of outages and in turn affects service level agreement.

Another complementary track of research explores the use of energy storage devices (e.g. UPS) to even out variability of renewable energy [12, 13]. However UPS is not suitable for medium to long power outages. Moreover these devices are expensive, have limited charge/discharge cycles and are not environmental friendly [14, 9].

A third direction is to leverage geographical diversity of internet-scale systems to provide environmental gains [5, 6, 15]. Specifically, these distributed systems dynamically route requests based on local renewable energy availability. However this technique works best for stateless, shortlived requests such as serving static content.

We think that in order to maximise the use of renewable energy we need to consider these observations. (*i*) Remote locations usually have higher renewable energy potential [9]. (*ii*) Geographically disjoint locations complement each others in terms of overall energy production [16]. (*iii*) Incorporating renewable energy within the electricity grid is costly and inefficient [17, 18, 19, 20].

This supports a more unconventional solution; we rely on *off-grid* architectures for integrating renewable energy into datacentre computing [21, 22]. Datacentres are located next to renewable energy sites and interconnected via dedicated links. Jobs are initially submitted to sites with adequate compute and power capacity, and are either migrated to another location or halted if the current capacity at the local site falls below the minimum level necessary to support them.

This paper explores, using a holistic model and case studies, the intrinsic trade-offs associated with this off-grid system design and quantifies the different overheads in provisioning the architecture and execution of computations.

Our results, based on two real-world case studies, have found that an optimised configuration consisting of ten distributed datacentres can fully rely on renewable intermittent energy to support computations with approximately 2% additional delay in their completion time relative to conventional architectures. However, this flexibility comes at the expense of up to 50% increase in the number of servers.

We start by describing this new architecture in detail (Section 2). We then formulate a board, encompassing model to inform a simulator for this type of architecture (Section 3). Based on this model, we present two case studies (Section 5 and 6) that illustrate the inherent trade-offs associated with provisioning datacentres and computing in this off-grid system. Finally we discuss related work (Section 7) and conclude (Section 8).

Our goal is to move the discussion of this off-grid architecture forward. Having a large problem in hand means that our model is necessarily an approximation. Nevertheless we believe that this work takes the first steps to identify the many factors inherent in this architecture and quantify their costs and benefits.

2 Architecture Overview

We present an example off-grid architecture (Figure 1) that enables the integration of renewable energy into datacentre computing without requiring the expensive investment necessary to deploy renewable energy at scale [21]. Datacentres are located next to renewable energy sites and linked together with a dedicated communication network. The

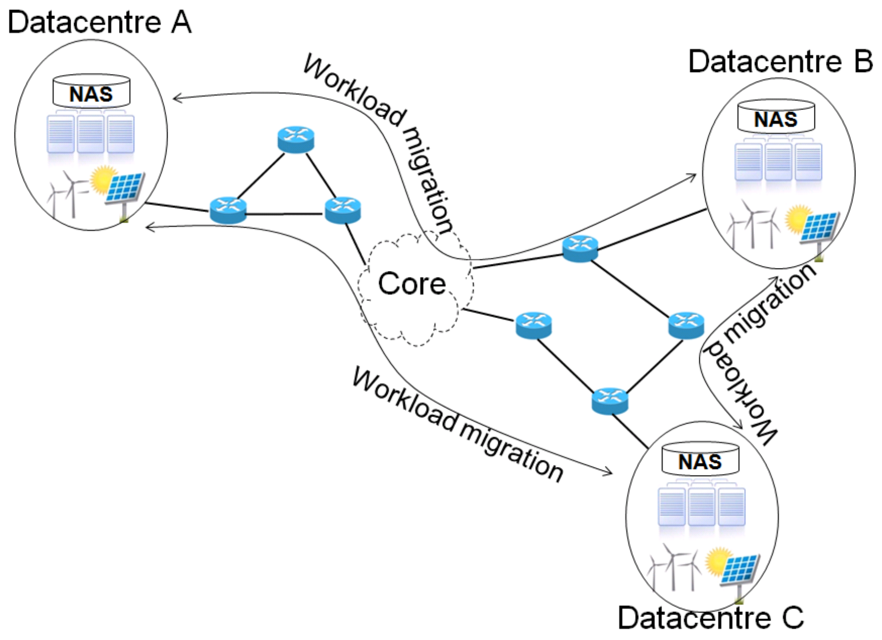

Figure 1: Off-Grid Computing Architecture [21]

system automatically migrates workloads (application and data) to locations that have enough power so that they continue execution despite the variability of renewable energy.

Several technologies support the realisation of this design. (*i*) Virtualisation (and more specifically live migration) is a practical technology, which enables the seamless mobility of virtual machines (VMs) to other physical hosts with minimal downtime. (*ii*) High-bandwidth optical systems provide quick transfer of state data during migrations. Additionally, (*iii*) scalable modular datacentre design permits the easy deployment of computing facilities at remote locations.

Computing using this architecture has technical hurdles, which have been addressed in the literature: (*i*) smart routing of jobs based on fluctuating energy levels [5] (*ii*) planning and provisioning support through predicting migration times, i.e. the duration of the migration process and the length of service interruptions [23, 24] and (*iii*) efficient disk-state synchronisation to remote destinations [25, 26].

Obviously, not all workloads are appropriate for inclusion in this model. In practice, batched compute-intensive jobs that accommodate slack scheduling are good candidates. However, latency sensitive jobs might fare poorly as they will be affected by service interruptions and propagation delays.

The Big Data Challenge: the biggest challenge that is typically associated with workload mobility is data synchronisation, especially for Big Data applications (e.g. MapReduce). However it has been demonstrated in a previous research that 80% of MapReduce jobs require only a fraction of the input data in the order of megabytes [27, 28]). Moreover 80% of jobs write small files (< 64 MB) to the distributed file system [29, 30]. Effectively, we assume that each location will have local access to this small highly accessible data. We also restrict the set of "movable" jobs to the ones that either have their data ready at the destination or require the transfer of small amounts.

There exist a few techniques in the literature that further reduce the amount of data to

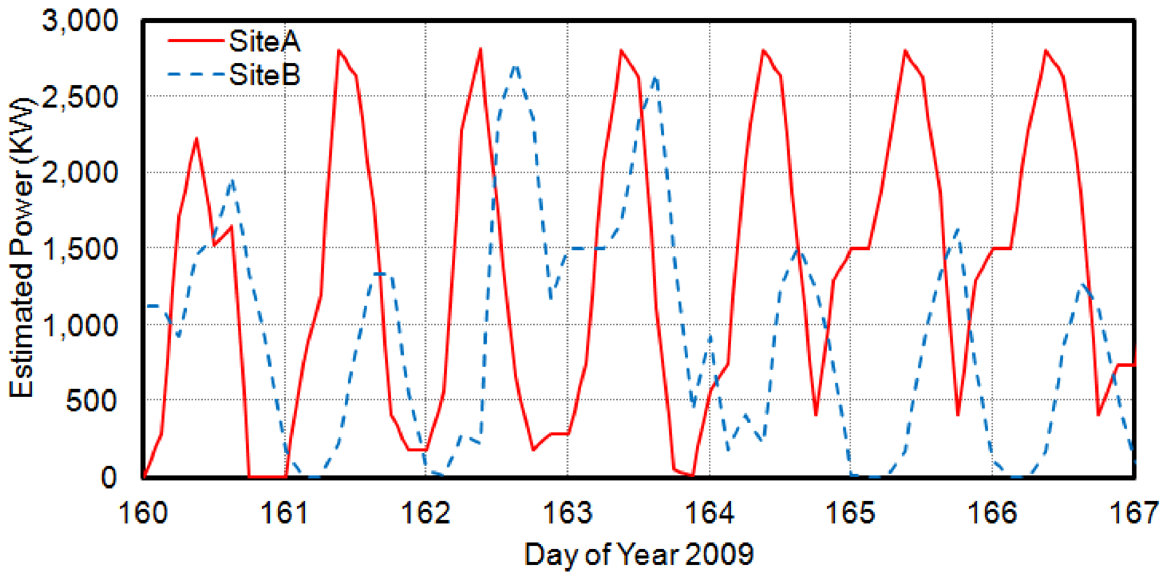

Figure 2: Estimated Power (Wind + Solar) for Two Sites Having Each 10,000 m^2 of Solar Cells and one 1.5 MW Wind Turbine.

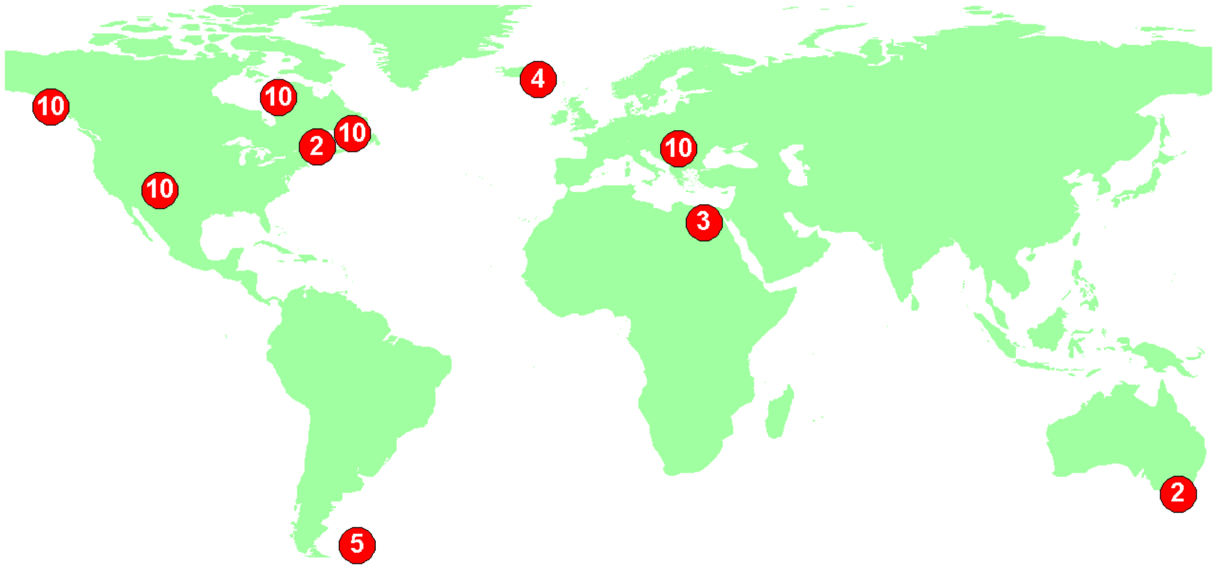

Figure 3: Datacentre Locations. Number indicates the maximum level of distribution in which a particular site participates.

be transferred. It has been observed that the long term average write rate of applications is low compared to gigabit links [23, 31] which makes the synchronisation process tractable. Moreover, smart write coalescing [23, 31] has shown to reduce the amount of data sent by absorbing transient overwrites. Coupled with selective data copying heuristics (e.g. based on access frequency) [26] and redundancy elimination techniques (e.g. deduplication) [32], the total disk-state transferred can be minimised.

3 Model Formulation

3.1 Datacentre Model

Datacentres consume energy to host IT equipment such as servers, storage nodes and network devices. Datacentres also have cooling and power distribution systems that support the operation of this equipment.

We model the energy usage of datacentres by considering the power of the computing infrastructure (servers) and the supporting systems (cooling and power distribution).

The off-grid infrastructure relies on virtualisation; i.e. one physical server can host several VMs. We assume that each physical machine (k) in the datacentre utilises $P_{mstatic}$ when it is on and a further P_{mdyn} for each active VM (1).

$$P_m(k) = P_{mstatic}(k) + \sum_{i=1}^{NumVM(k)} P_{mdyn}(i) \tag{1}$$

Based on previous studies we set $P_{mstatic}$ and P_{mdyn} to 150 W and 25 W respectively [33, 34] and assumed that up to four medium-sized VMs can run at one physical machine.

We estimate the additional overheads of each datacentre (n) due to power distribution and cooling using the power usage efficiency (PUE) metric (2).

$$P_{dc}(n) = \mathrm{PUE}(n) \times \sum_{k=1}^{NumMachines(n)} P_m(k) \tag{2}$$

The PUE was set to 1.1 (i.e. 10% of the total energy is consumed by the cooling and power distribution systems), which is currently achievable in the industry [35, 36]. With “free” cooling and higher datacentre operating temperature becoming mainstream, many locations can potentially reach this low PUE with manageable effect on system reliability [37].

In our model, we do not consider overheads that might be associated with dynamic server provisioning techniques (e.g. turning on/off servers) or with chillers for ramping up-/down cooling. It is expected that these overheads will continue to fall [4] to insignificant values.

Moreover, we do not consider networking infrastructure inside one datacentre in this work as datacentre networks (whether it is conventional or fat tree) accounts for a small (5%–10%) percentage of the overall facility power [38].

We acknowledge the fact that PUE changes according to the outside environmental conditions. However we ignore this complexity for a first order analysis. Moreover our model is easily extensible to accommodate a dynamic PUE.

3.2 Network Model

In this system, datacentres are interconnected with dedicated links that will carry memory and disk-state during synchronisations. Consequently, the transition of a workload to another physical location consumes energy in the processing of packets/frames by different components of the network. We use an existing network power model, which focuses on the core network and the underlying optical transport systems [39].

3.2.1 Core Network

The core network consists of a few large routers that route traffic to neighboring nodes. We assume that routers fully process IP packets. Consequently, the power incurred by a packet in the core network (P_c) is directly proportional to the number of hops ($numHops$) that is required for the packet to reach its destination along with the power of each router (P_r). Additionally, core routers are usually provisioned for redundancy and future growth factor ($factor_r$) (3). We assume that the network is energy-proportional (a promise of future photonic systems) and hence the power values represent only the fraction required to support migration based on the amount of data transferred.

$$P_c = (numHops + 1) \times P_r \times factor_r \tag{3}$$

We used the power profile of a single-shelf 16-slot Cisco CRS-3 core router that consumed a maximum of 13.2 KW for the processor, line cards and switching fabrics, and had a switching capacity of 4.48 Tbps [40]. Additionally, a redandancy/growth factor of 8 was considered according to published numbers [39]. This translates to an energy profile of approximately 2.7 nJ/bit.

3.2.2 Optical Transport Systems

Long-haul networks between core routers employ wavelength division multiplexing (WDM) transport systems that require high-performance modulations of optical signals. There are essentially two similar kinds of WDM links: terrestrial and undersea links. Both systems require line terminals (P_{tm}) at each end in addition to signal amplification (P_a) every $AmpDist$ distance. A redundancy/future growth factor is also considered ($factor_t$). The total power used by the transport systems (P_t) is:

$$\begin{aligned} P_t = &\left(\frac{Dist(ter)}{AmpDist(ter)} \times P_a(ter) + 2 \times P_{tm}(ter)\right) \times factor_t(ter) \\ &+ \left(\frac{Dist(sea)}{AmpDist(sea)} \times P_a(sea) + 2 \times P_{tm}(sea)\right) \times factor_t(sea) \end{aligned} \tag{4}$$

In our simulations, terrestrial systems used Fujitsu Flashwave 7700 line terminals that consumed 811 W for 44 channels (40 Gbps per channel). An amplifier was required every 100 km which drew 622 W. Similarly, undersea systems used 4.5 KW Fujitsu Flashwave S650 line terminals for 64 channels (10 Gbps per channel) and 50 W repeaters every 50 km [39]. We assumed that 70% of the link distance was undersea and the rest on land. This adds up to approximately 29.8 nJ/bit with a redundancy/future growth factor of 4.

3.2.3 Propagation Delay

While overheads in communication networks depend on processing, queueing and propagation delays, only propagation delays are considered in our model as they represent the minimum possible overhead that can be achieved. We therefore used the speed of light, its refraction index traveling in glass and the direct distance between any two datacentres to estimate the minimum time required for a packet to be transferred between these two locations. While this assumption may appear optimistic and currently unrealistic, we envision that future networks would have the capacity to assign a dedicated light path in a WDM link for the purpose of workload migration which would have virtually zero interference with the normal inter-datacentre traffic.

3.3 Renewable Energy Representation

We focus on wind and solar energy as they are prevalent but intermittent in nature. We represented them based on weather station readings for the year 2009.

We extrapolate wind speed from the weather station readings [41] at ground level to 80 m, which is the typical height of the wind turbine. The wind energy is then modelled as a function of the turbine power curve [16].

Solar irradiation on the other hand is a factor of latitude, time, atmospheric conditions and aerosols. We estimate the amount of solar energy using a simple insolation model [42] that incorporates these factors to approximate clear-sky solar irradiation at a given location. We also account for cloud cover available from the weather station readings [41].

Weather station readings are reported on three-hour interval; however this resolution is not enough for the purpose of our simulations. Therefore, linear interpolation was used between the actual values as an approximation for a higher granularity.

In our simulations, we assumed that each datacentre had 10,000 m^2 of solar cells with an average efficiency of 13% and one 1.5 MW wind turbine. The provisioning of wind and solar energy harvesting equipment was held constant. This choice was based on the facts that: (*iv*) renewable energy is intermittent which renders the planning process difficult (and beyond the scope of this research), (*v*) the amount of computations that can be executed is already capped by the computing capacity provisioned despite how many wind turbines or solar cells are available on site, and (*vi*) most importantly this setup gives roughly equal weight to wind and solar contribution in our simulations.

We believe that this set of energy harvesting equipment is good enough to illustrate the behaviour of the system. It is not too small to restrict the number of computations that can get executed and it is not too large to mask the effect of renewable energy intermittency.

Figure 2 illustrates the power output for two different sites during one week in July 2009. The periodic effect of solar power on the total output is clear from this figure. Moreover due to the time zone difference between the two locations, the peak power output is shifted. In other words the two sites complement each other. This is exactly the opportunity that is exploited by relocating workloads to where energy is currently available which enables sustaining the computation demand despite the intermittency of renewables.

3.4 Architecture Overheads Model

There are different overheads inherent in this architecture related mainly to workload migration and intermittency of renewable energy. Generally these overheads fall either into (*i*) service interruptions or (*ii*) additional data transfer and, by extension, energy consumed in the network. Service interruptions are caused by either migration originated downtime or lack of resources. Moreover the migration process requires synchronisation of memory and disk-state to the destination which affects the network in terms of data load and energy.

3.4.1 Service Interruptions

The impact on availability inherent in the system comes from two factors: (*i*) during the final stop-and-copy migration phase and (*ii*) when the application is paused due to lack of resources.

Migration Originated Downtime: the duration of the stop-and-copy stage of the migration process is workload, network and machine specific [23, 43]. Generally we estimate this migration originated downtime based on the amount of state that needs to be transferred on a given link speed plus the propagation latency. We assumed that all associated VMs with a given job are relocated to the destination in parallel. In other words, the total migration originated downtime per job is equal to the downtime for a single VM under this specific job multiplied by the number of migrations.

Availability of Resources: sometimes there are not enough resources, in terms of compute and power capacity in the system, to accommodate all jobs. Therefore a number of jobs (and their VMs) at each of these events (e) have to be paused until more resources are available. Execution delay for a given job is therefore modelled as:

$$\begin{aligned} TotalDowntime(job) = & \sum_{e=1}^{NumEvents(job)} PausingTime(e) \\ & + NumMigration(job) \times VMDowntimeMigration(job) \end{aligned} \tag{5}$$

3.4.2 Data Transfer

There are two components contributing to the amount of data that needs to be transferred during migrations: (*i*) memory-state synchronisation and (*ii*) disk-state synchronisation. The pre-copy migration model iteratively transfers memory-state modifications to the target host with the expectation to have a short final stop-and-copy phase [44]. On the other hand, storage is "smartly" synchronised so that when migration is triggered disk-state difference is small enough to enable a minimal downtime [25]. Note that applications are always migrated with their associated disk data.

While it seems that this process is less flexible due to the constraint that a VM can only be migrated within the set of datacentres in which its disk image is synchronised, in practice it is expected that technological (e.g. multicast streaming) and organisational (e.g. pre-computed VM scheduling policies) techniques will mitigate this requirement. Additionally destinations can be pre-selected with the help of weather forecasts.

The amount of memory-state is estimated to be equal to the memory size allocated per VM multiplied by a factor that accounts for the working set size. For one migration event per job, data is aggregated for all its associated VMs running at that point. A job can have several migrations, which of course increase the total data sent. Even if a job does not require migration, it still consumes bandwidth by the proactive disk-synchronisation mechanism. For storage synchronisation, the disk data sent by a specific job is equal to its running time multiplied by the associated number of VMs and their average disk write rate.

The total data transferred by a given job is:

$$\begin{aligned} TotalData(job) = & \sum_{i=1}^{NumMigr(job)} \sum_{j=1}^{NumVM(i)} MemSize(j) \times Factor \\ + & \sum_{t=1}^{Runtime(job)} NumVM(t) \times AvgVMWriteRate(t) \end{aligned} \tag{6}$$

4 Evaluation Methodology

The off-grid infrastructure is evaluated by comparing the amount of completed computations and their runtime with respect to the *traditional centralised case*, which is a configuration that constitutes one datacentre powered by the electricity grid and has on-site generators as a backup. Consequently, it is guaranteed that this datacentre has a stable supply of power that can be used to run the computation demand. This conventional configuration (i.e. the provisioned computing capacity along with the number of completed computations and their runtime) is used as the reference point in the analysis.

4.1 Metrics

4.2 Site Selection

The locations at which datacentres are built play an important role in minimising the total cost of ownership. Conventional site selection strives to optimise different parameters such as energy cost, connectivity and proximity to clients. However, maximising renewable energy output is our primary focus as we assume that other criteria (e.g. connectivity) would be easier to achieve.

The infrastructure is distributed by design over different geographical regions. We chose fifty locations according to their wind and solar power potential (Figure 3). These sites are picked to complement each other in terms of aggregate power generated: (*i*) their time zones are far apart to maximise solar power over a 24 hour period, (*ii*) their locations are chosen in different hemispheres so that winter and summer interchange and (*iii*) their locations are well distributed to even out intermittency of wind and solar power [16]. With many datacentres now being operated at higher temperatures, hot climate is not eventually a concern at these chosen locations [45].

		Slack Reserves					
		0%	10%	20%	50%	100%	200%
Stable Power	0%	N/A	5	5	4–5	4–10	4–10
	10%	5	4–5	4–5	3–5	2–10	2–10
	20%	4–5	3–5	2–5	2–5	2–10	2–10
	50%	2–5	2–5	2–5	2–5	2–10	2–10
	90%	2–5	2–5	2–5	2–5	2–10	2–10
		January 2009					

		Slack Reserves					
		0%	10%	20%	50%	100%	200%
Stable Power	0%	3–5	3–5	3–5	3–5	3–10	3–10
	10%	3–5	3–5	3–5	2–5	2–10	2–10
	20%	2–5	2–5	2–5	2–5	2–10	2–10
	50%	2–5	2–5	2–5	2–5	2–10	2–10
	90%	2–5	2–5	2–5	2–5	2–10	2–10
		July 2009					

Table 1: Level(s) of Distribution Executing the Target Computations with **No Bound on Average Increase in Execution Time Relative to the Centralised Case** (x–y Means Between x and y)

		Slack Reserves					
		0%	10%	20%	50%	100%	200%
Stable Power	0%	N/A	N/A	N/A	N/A	10	5–10
	10%	N/A	N/A	N/A	4–5	4–10	3–10
	20%	5	4–5	4–5	3–5	2–10	2–10
	50%	2–5	2–5	2–5	2–5	2–10	2–10
	90%	2–5	2–5	2–5	2–5	2–10	2–10
		January 2009					

		Slack Reserves					
		0%	10%	20%	50%	100%	200%
Stable Power	0%	N/A	N/A	N/A	N/A	5–10	5–10
	10%	N/A	5	5	4–5	3–10	3–10
	20%	5	4–5	3–5	3–5	2–10	2–10
	50%	2–5	2–5	2–5	2–5	2–10	2–10
	90%	2–5	2–5	2–5	2–5	2–10	2–10
		July 2009					

Table 2: Level(s) of Distribution Executing the Target Computations with < **10% Bound on Average Increase in Execution Time Relative to the Centralised Case** (x–y Means Between x and y)

4.3 Job Placement Algorithm and Model Assembly

We assume that migrations happen at the job level to avoid inter-process communications crossing the datacentre boundary which will otherwise degrade performance significantly due to geographical latency.

A greedy placement and scheduling algorithm is used in our simulations. When a new job arrives to the system, the algorithm searches for a datacentre that has enough compute and power capacity to accommodate it (Eq. 2). If no datacentre is found meeting these requirements, execution of this job is queued until resources are available.

In the situation where power drops at one site, some jobs have to be evacuated to different available datacentres. The algorithm searches for locations that have enough resources to host any of these jobs. Otherwise, they are paused and resumed as soon as possible. Priority is given to running and paused jobs over new ones for a given class.

We compute service interruptions that are expected to happen due to transient lack of resources and migration originated downtime (Eq. 5). Moreover during the migration process, state information is synchronised to the destination (Eq. 6). This data in the form of packets sent on the link consumes energy at the core (Eq. 3) and transport (Eq. 4) networks.

5 Case Study A: Cluster-type Jobs

5.1 Google Cluster Workload Trace

The first case study involves a computation demand driven by a Google cluster trace [46]. This dataset is collected over a period of 1 month (May 2011). It includes details about jobs and their associated compute tasks from a 11,000-machine cluster. It is believed that they represent a variety of back-end tasks executed on Google compute clusters (e.g. compute-intensive web logs analysis, user-facing web search processing and other MapReduce workers) which have diverse service level requirements for throughput, latency and jitter [47].

The dataset contains timestamped information about jobs representing the different states of execution (i.e. submit, schedule, evict, fail, finish, kill and lost). We only extracted details about tasks that have completed successfully. Effectively we captured scheduling events for these tasks and allocated them according to the process described in Section 4.3. We followed job priorities available in the trace. Moreover, we assumed that each machine can host up to four tasks (encapsulated each in a VM). We also scaled up the number of jobs by two to correspond to the scale of our simulations.

5.2 Infrastructure Provisioning and Planning

Based on the datasets and models discussed in Section 3, the amount of computations executed (and their runtime) were evaluated for different configurations and compared to the traditional centralised case. Our analysis focused on (*i*) the level of distribution (the number of datacentres), (*ii*) the amount of computing capacity provisioned for each datacentre and (*iii*) the percentage of non-intermittent load at each site. We present results for January and July 2009.

		Slack Reserves					
		0%	**10%**	**20%**	**50%**	**100%**	**200%**
Stable Power	**0%**	N/A	N/A	N/A	N/A	10	10
	10%	N/A	N/A	N/A	N/A	5–10	4–10
	20%	N/A	N/A	N/A	5	4–10	4–10
	50%	N/A	4	4–5	2–5	2–10	2–10
	90%	2–5	2–5	2–5	2–5	2–10	2–10
		January 2009					

		Slack Reserves					
		0%	**10%**	**20%**	**50%**	**100%**	**200%**
Stable Power	**0%**	N/A	N/A	N/A	N/A	10	10
	10%	N/A	N/A	N/A	N/A	4–10	4–10
	20%	N/A	N/A	N/A	5	3–10	2–10
	50%	5	2–5	2–5	2–5	2–10	2–10
	90%	2–5	2–5	2–5	2–5	2–10	2–10
		July 2009					

Table 3: Level(s) of Distribution Executing the Target Computations with < **1% Bound on Average Increase in Execution Time Relative to the Centralised Case** (x–y Means Between x and y)

5.2.1 Level of Datacentre Distribution

We start by considering the level of distribution (i.e. the number of datacentres in the configuration). This allows intermittent power output from different geographical sites to complement each other. The level of distribution in the architecture was varied at 2, 3, 4, 5, 10, 25 and 50 datacentres. We selected each subset according to the same rules (discussed in Section 4.2). Figure 3 shows the locations of these subsets (> 10 omitted from the figure for clarity). The amount of machines provisioned at each site was initially set to the total machines of the conventional configuration (i.e. 15,000 machines) divided by the level of distribution (i.e. the number of datacentres). As the number of datacentres was increased, power variability was better smoothed out. However, this usually resulted in more machines being provisioned at each site.

5.2.2 Slack Reserves

As servers are spread over different sites, the system requires additional computing capacity. More machines per datacentre are needed to execute the local computations as well as the migrated ones from other locations. This is due to the fact that not all sites are available at the same time because of renewable energy intermittency.

This additional computing capacity is defined as slack reserves. This value was varied at 0%, 10%, 20%, 50%, 100% and 200% of extra machines which provide sensible Capex overheads for operators.

For low levels of distribution (e.g. 2 and 3), there is not enough location diversity to even out intermittency and sustain the target computation load. For high levels of distribution (e.g. 25 and 50), even large slack reserves of 200% was insufficient to cope with the load. Dividing up the same number of machines on much more locations (powered by intermittent sources) might lead to a less available configuration as the compute capacity at each node decreases considerably.

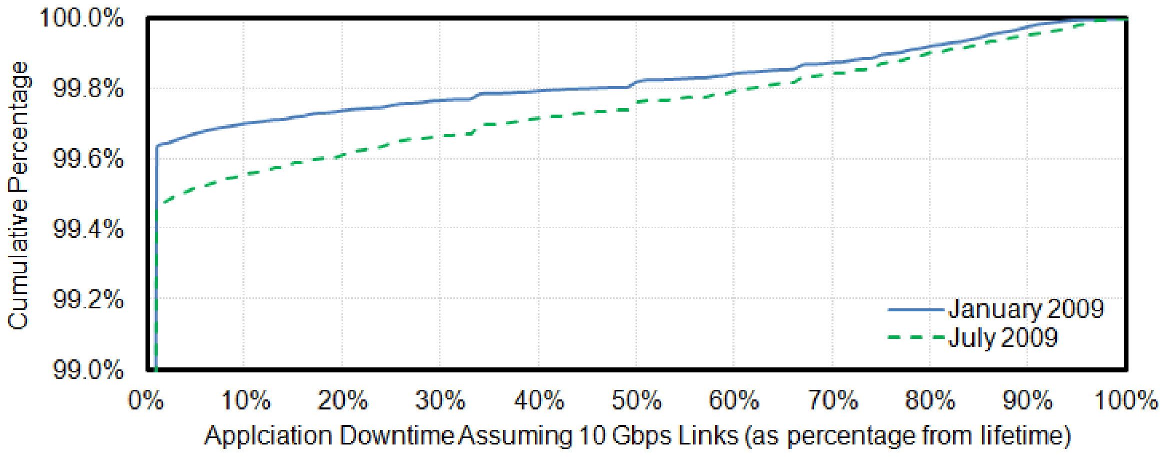

Figure 4: Cumulative Percentage of Application Total Downtime (Percentage of Lifetime, VM Link Speed= 10 Gbps, VM Working Set Size= 0.5 GB)

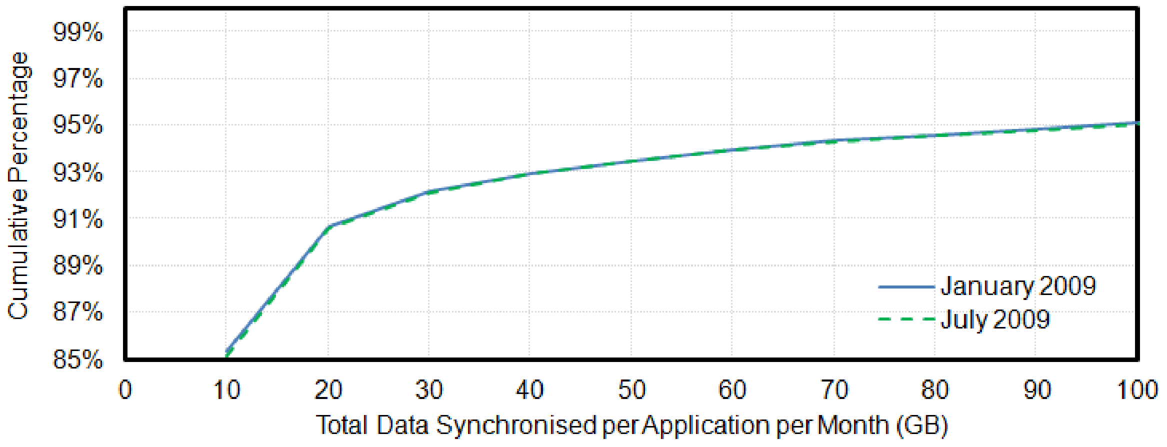

Figure 5: Total Data Synchronised per Application per Month (in GB, VM disk write rate= 0.5 MB/s, VM memory size= 7.5 GB)

5.2.3 Power Mix

For low level of distribution, there is not enough location diversity to smooth out renewable energy intermittency and, by extension, jobs have to be delayed. Rather than increasing the level of distribution (and suffering more slack reserves), one can rely on a power mix that has a non-intermittent component guaranteeing a baseload. This mix would enable the timely execution of computations while minimising slack reserves.

The non-intermittent power load at each site is parameterised as a percentage of the maximum power needed for the provisioned computing capacity. The chosen values in our simulations were 0%, 10%, 20%, 50% and 90% for the datacentre. Although it might be impractical to consider 0% stable load for datacentres, it provides an upper bound on the amount of machines required.

Executing the target number of computations is not the only metric associated with performance in our evaluation. In fact we are also interested in the runtime of jobs; i.e. whether they incur considerable delays during their life cycle.

5.2.4 No Bound on Additional Execution Time

Table 1 shows the level of distribution required for possible configurations of different slack reserves and non-intermittent baseload that execute the target number of computations; but with no bound on the increase in execution time per job relative to the centralised case. In fact, we have observed more than 60% increase in runtime in some of these configurations.

In these configurations, a moderate level of distribution (5) is often required to minimise both the slack reserves and the non-intermittent baseload. On the other hand, we can still rely on 2 or 3 datacentres but with considerable increase in slack reserves and baseload. We effectively need to provision the system for almost the maximum load at each location because we do not have enough geographical diversity to smooth out intermittency of power.

5.2.5 10% Bound on Additional Execution Time

We then restrict the additional increase in average execution time per computation relative to the centralised case to a maximum of 10%. It represents situations where we deal with applications that require a certain bound on completion but at the same time have some slack in scheduling which can be exploited to follow availability of power.

Table 2 illustrates configurations that fulfil this requirement (i.e. 10% bound). Obviously, the set of configurations shrinks compared to the unbounded case. We have to moderately increase the slack reserves and the baseload with a level of distribution of 5 or more to achieve this target.

5.2.6 1% Bound on Additional Execution Time

We further constrain runtime: $\leq$ 1% additional increase in execution time compared to the centralised case. It presents scenarios that require strict deadlines with almost no slack in scheduling.

Table 3 shows that a considerable increase in machine capacity or baseload is often required. For example, 10 distributed datacentres with at least 100% slack reserves are needed for the unaffected execution of computations.

The system can still be configured with a modest amount of slack reserves and a limited increase in the non-intermittent baseload. For example, a configuration provisioned with 20% non-intermittent load, 50% additional machines and a moderate distribution level (e.g. 5) has the capacity to execute the target number of computations.

We note that there are slight differences between the amount of computations done in January and July. Power output is not exactly the same because of different weather conditions. However, the general trend is similar.

5.2.7 Optimal Configurations

The previous section showed that resources can be optimised by having a moderate increase in machine capacity along with a small load of non-intermittent power. This section discusses other configurations that improve on the previous results. In these cases, each datacentre can have *different computing capacity and baseload values.* They all achieve the target number of computations with less that 1% additional increase in execution time per job relative to the centralised case.

The placement algorithm was modified to accommodate that datacentres have different sizes. The algorithm assigns jobs that require less machines to datacentres that have less computing capacity. In doing so, resources are freed for the bigger jobs (that can not be accommodated in the smaller datacentres anyways) to run in the larger datacentres.

Configuration X: 100% Renewable Energy Having four big datacentres (4,000 servers each) with six smaller ones (1,000 servers each) enables the timely execution of computations using 100% wind and solar energy. As the level of distribution is 10, sites complement each other in terms of energy. This comes at the expense of 47% additional computing capacity.

Configuration Y: No Slack Reserves A set of three more reliable datacentres within the infrastructure having each 4,000 servers and 500 KW of non-intermittent load, and another three datacentres with 1,000 servers each can achieve the target amount of computations. This setup does not have any increase in the computing capacity over the amount of the centralised case. Additionally, the stable power component is just 37% of the total. Configuration Y is ideal when the cost of servers is dominant and hence the number of machines should be minimised.

Configuration Z: Hybrid Design Configuration Z reduces the amount of non-intermittent load to below 8% (150 KW for the three big datacentres) and has a 40% increase in computing capacity. This achieves a trade-off between the provisioned computing capacity and the non-intermittent load at each site. The purpose of this configuration is to optimise the values of all parameters in the system.

5.3 Overheads Associated with the Architecture

This section substantiates the various overheads associated with the infrastructure due to relocating workloads to different datacentres. We use an extreme configuration, which fully relies on wind and solar energy: four big datacentres having each 4,000 servers and six smaller datacentres having each 1,000 servers. This comes though at the expense of 47% additional computing capacity.

5.3.1 Service Interruptions

As we have discussed before, the impact on availability inherent in the system comes from either occasional lack of resources or migration downtime. We observed that pausing time caused by lack of resources elsewhere has a much bigger effect on total downtime than

Description	Non-intermittent Power	Additional Capacity
Configuration X 4 datacenters |4,000 servers each + 6 datacenters |1,000 servers each	0%	47%
Configuration Y 3 datacenters (500KW) |4,000 servers each + 3 datacenters |1,000 servers each	36.9%	0%
Configuration Z 3 datacenters (150KW) |4,000 servers each + 1 datacenters |3,000 servers each + 6 datacenters |1,000 servers each	7.8%	40%

Table 4: Optimal Configurations Guaranteeing the Execution of the Target Amount of Compactions

migration originated interruptions. In fact, more than 98% of the sum of total downtimes in our simulations can be attributed to pausing time related to lack of resources.

Figure 4 illustrates the cumulative percentage of application downtime (presented as a percentage of its lifetime). The lifetime of an application is the duration from the moment a specific job is submitted to the system until it is completed successfully. Hence it includes running, pausing and migration stop-and-copy times. The figure shows that more than 99.95% of the applications experience a total downtime less than 1% of their lifetime. In other words, service interruptions in the system are minimal apart from a very small set of applications.

The number of migration events required when power drops influence the total downtime endured by a specific application. We observed that more than 90% of the computations do not require migrations. These are either short-lived applications that get completed before power drop or small-sized applications that can run with a fraction of the resources.

On the other hand, 10% of all applications incurred at least one migration event. In fact 99% of these are migrated under 10 times. However, there are some applications (typically long-lived) that are relocated many times: around 150 migration events per month for the worst case. This is expected as the system fully relies on intermittent renewable energy.

5.3.2 Data and Energy Overheads

During migration, memory and disk-state are being transferred to the destination. Packets sent over the network consume energy which is considered as an overhead. This should be minimal compared to the energy used in driving the actual computations. Note that in our simulations a job (application) runs many *tasks each encapsulated in a VM.*

We explored the amount of state information that is synchronised per job assuming VM disk write rate equal to 0.5 MB/s (based on average values from a previous study [25]) and VM memory size equal to 7.5 GB similar to an Amazon EC2 large instance ($factor = 1.5$ in Eq. 6). Figure 5 shows that 90% of the applications transfer just over 20 GB of data

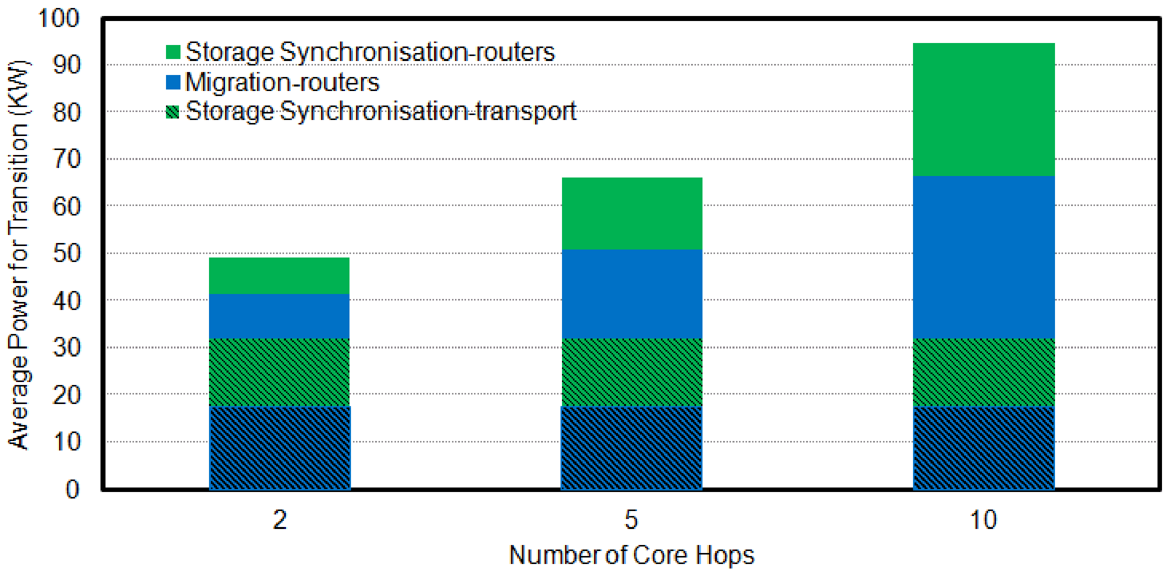

Figure 6: Average Power Required for Synchronisations for Different Core Hops (VM Disk Write Rate= 0.5 MB/s, VM Memory Size= 7.5 GB)

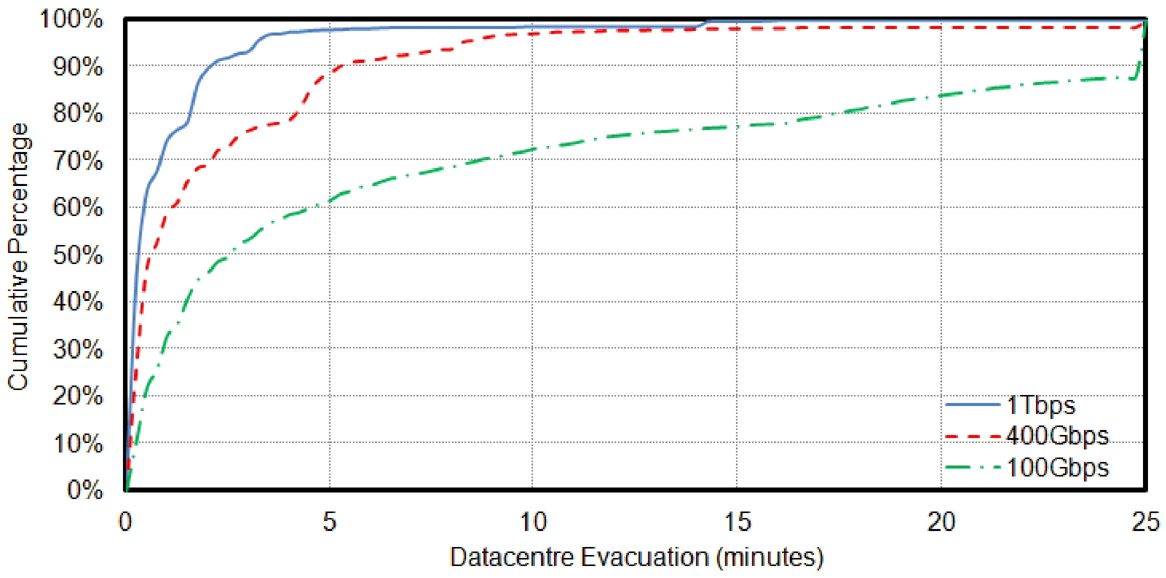

Figure 7: Datacentre Evacuation for a Specific Case (VM Link Speed= 10 Gbps, VM Working Set Size= 0.5 GB, VM Disk Write Rate= 0.5 MB/s, VM Memory Size= 7.5 GB)

per month. However, there are a few big jobs that send more than 10 TB per month because they run a lot of tasks or require many migrations.

Based on the network model that is presented in Section 3.2, the amount of data transferred by all jobs and their associated tasks is used to quantify the average power required for the transition. It is the power consumed by the communication network for the purpose of workload migrations. Figure 6 shows this average power classified by memory-state and disk-state, and includes both the power needed at the routing layer (i.e. core routers) as well as the WDM transport systems. Various hop counts are shown to represent different network configurations. We varied the core hops at 2, 5 and 10 to cover major configurations [48, 49].

For hop count 2, 5 and 10, the average power required to support migrations is around 65 KW, 90 KW and 125 KW respectively. The average power used in the routing layer is proportional to the number of hops due to the fact that packets are processed by a smaller number of routers. On the other hand, power required in the optical transport systems is fixed because it is dependant on the average distance in the test configuration. The total network power is evenly attributed to memory and disk-state synchronisation. We observed that the network power required to support migration is at least two orders of magnitude less than the total power generated from renewable sources in the configuration.

5.3.3 Datacentre Evacuation

When power drops at one site, it is expected that hundreds of machines need to be migrated to other locations and must share the links. This evacuation process is affected by the bandwidth of interconnects.

Evacuation is defined at the time required to migrate *all* applications that exceed the "current" energy capability of a specific datacentre. The system assumes that datacentres are interconnected using links dedicated to memory-state and disk-state synchronisation. As the storage synchronisation algorithm transfers proactively modified sectors to the destination [25], this process utilises a significant bandwidth, which was computed during simulations and factored out from the total interconnect capacity. The remaining bandwidth was then available for migrations and affected the duration of the evacuation process. Time for each individual migration accounted for both the pre-copy and final stop-and-copy phases.

Three different interconnect speeds were chosen: 100 Gbps, 400 Gbps and 1 Tbps. The first two represent foreseeable deployments [50] while the latter is a reasonable future prediction. In fact, there are already some experiments that have successfully demonstrated transmission speeds in the order of ten terabits per second over a single fiber [51].

Figure 7 illustrates the cumulative percentage of the time required for datacentre evacuations in the case that each VM has access to 10 Gbps interface, has 7.5 GB memory ($factor = 1.5$ in Eq. 6) and its disk write rate is 0.5 MB/s. For 1 Tbps interconnect, the 90% and 97% of the evacuations happen in less than 2 minutes and 3 minutes respectively. In the case of 400 Gbps, durations for evacuation increase by a factor of 2. Even at 100 Gbps, delays can be tolerated at 15 minutes for 90% of evacuations. These results prove that (i) migrations in the off-grid infrastructure happened in a timely manner and (ii) support for this transition had no severe impact on the planning of datacentre operation especially for high-speed interconnects.

6 Case Study B: MapReduce Jobs

6.1 Facebook Workload Trace

As the work presented in the paper relies on trace-driven simulations, it was important to present a second case study that would give the reader some assurance about the results.

The second case study involves a specific workload type: MapReduce tasks. There is currently wide-spread interest in the MapReduce paradigm for large-scale data processing and analysis [52]. Programs written according to this model are automatically executed in parallel and are tolerant to faults on large scale clusters. MapReduce is being used in web search, sorting, data mining and machine learning [53].

We used MapReduce job traces from the primary Facebook production 3,000-machine Hadoop cluster. The length of the trace is 6 weeks from Oct. 1 to Nov. 15, 2010 and contains details about roughly a million jobs. Records in the trace provide input/shuffle/output sizes, start/finish times, the number of tasks in a given job among other detail [27].

We scaled up the number of jobs to correspond to the size of our simulations. Each physical server can host up to two Map and two Reduce tasks. Power and network provisioning is the same as the first case study.

6.2 Computation Delay

In this case study, we compared a test configuration that fully relies on renewable energy and has ten distributed datacentres with the centralised case. The amount of slack reserves for this test configuration was varied in our simulations.

Specifically, we studied the average additional delay per job (as a percentage of its respective runtime in the centralised case) that was experienced for different computing capacities. Figure 8 illustrates this metric for each month of the year 2009. The average delay per month for 0%, 20% and 50% slack reserves ranges between 32%–96%, 6%–20% and 2%–6% respectively. We have found out that configurations with no slack reserves are less tolerant to variability of renewable energy (as illustrated by the spikes in the figure).

We estimated the amount of data that needs to be transferred during migrations. When migrating Map tasks, Map input data has to be locally available. Additionally if the Map tasks belong to jobs that load data, output data has also to be transferred as it would be used by subsequent tasks. In the case of migrating Reduce tasks, shuffle data has to be copied to the destination. Additionally, we assumed no partial transfer; i.e. regardless of the job progress all its input data was transferred as a requirement for migration.

We have observed that in order to support migration for MapReduce jobs in the test configurations, a 250 Gbps interconnect between datacentres is required. This is an average value computed by analysing the total input data transferred by all jobs per month for the purpose of migration and assuming uniform distribution. We think this is a reasonable overhead considering future interconnects.

There could be arguments advocating to pause computations and resume them later when power is available (as opposed to migrating computations to another datacentre where they can continue execution). In these "no-migration" cases, the system design is

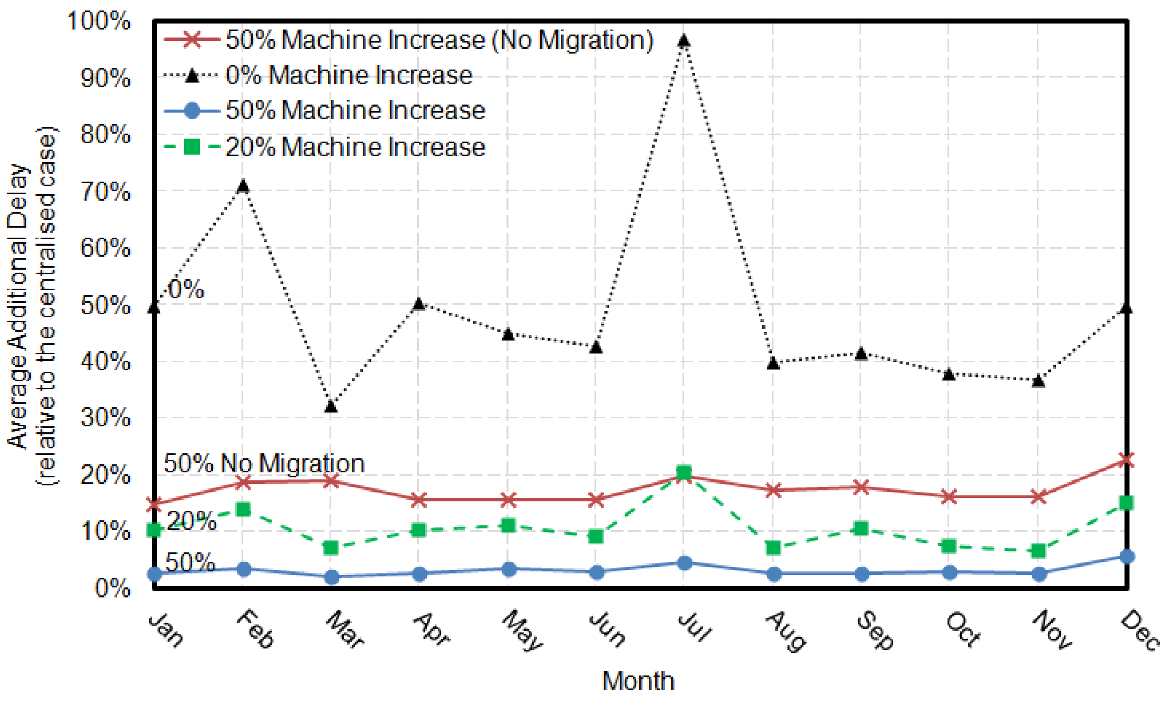

Figure 8: Average Additional Delay per MapReduce Job per Month for Different Additional Computing Capacities Relative to the Centralised Case

obviously much simpler and does not incur overheads associated with migrations. However, this might come at the cost of additional delays in computations.

We compared the effect of having a configuration (50% slack reserves) with and without migration on average additional job delay. The average delay per job increases from 3% to 17% when migration is not employed (Figure 8). Although the traces are dominated by short-lived jobs [27], there are a few long-lived jobs that experience significant delays during their executions due to intermittency of renewable energy.

7 Related Work

There are a few approaches that use renewable energy at the inter-datacentre scale. Liu et al. provide distributed algorithms that compute the optimal routing decisions for internet-scale systems which can be leveraged to route requests to areas where green energy is available [5]. Similarly, Le et al. discuss cost-aware load migration and placement across different geographical datacentres which take into account all electricity-related costs as well as transient cooling effects [6, 15]. Chiu et al. propose to use low-cost workload migration to integrate renewable energy in the electricity grid.

At the intra-datacentre scale, Deng et al. discuss the datacentre design trade-offs considering wind and solar resources [8]. Also, GreenSlot is a parallel batch job scheduler for datacentres which automatically selects the cheaper energy sources in order to reduce the overall cost [7]. Similarly, Gmach et al. illustrate an approach for matching power that may come from the grid, on-site renewable sources and energy storage sub-systems with the computing demand in datacentres [11]. They use power capping techniques at different levels of the datacentre to control demand with minimal impact on application performance. Using historical traces, their study estimates the amount of conventional power needed to smooth out renewable energy variability. Singh et al. implemented a system that can help maintain server availability in renewable energy-powered datacentres [14]. Their design relies on a downtime-bounded migration algorithm that can enable committing changes before the power is cut off a specific server.

Another complementary track of research explores the use of energy storage devices (e.g. UPS) to even out variability of renewable energy [12, 13]. Ren et al. provide a cost-based study on many of the above technologies [9].

8 Conclusions

This paper evaluated the viability of integrating remote renewable energy in datacentre computing using an off-grid architecture. We formulated a holistic model of this architecture. We then presented two case studies based on traces from Google and Facebook that quantified the associated overheads.

We discussed the challenge of provisioning this unconventional architecture: how datacentres can be distributed to enable the timely execution of computations despite renewable energy intermittency but with a moderate increase in machines. In fact there is an intrinsic trade-off between the level of distribution and the amount of slack reserves required in the design. We need more distributed datacentres to even out variations in power generation which comes at the cost of adding more machines to support the execution of applications as not all sites will be up at the same time.

We have found that a geo-distribution configuration provisioned with 50% additional computing capacity can executed the target amount of computations with $\approx 2\%$ delays.

9 Acknowledgments

We are grateful to Yanpei Chen for his insights and comments. We thank Facebook and Dhruba Borthakur for giving us access to the traces used in the second case study. We thank also Jon Crowcroft and Toby Moncaster for their feedback.

References

[1] J. G. Koomey, "Growth in Data center electricity use 2005 to 2010," Tech. Rep., 2011.

[2] K. G. Brill, "The invisible crisis in the data center: The economic meltdown of Moore's law," Tech. Rep., 2007.

[3] E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath, *Dynamic cluster reconfiguration for power and performance*, 2003.

[4] D. Meisner, C. M. Sadler, L. A. Barroso, W.-D. Weber, and T. F. Wenisch, "Isca," 2011.

[5] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew, "Greening geographical load balancing," in *SIGMETRICS*, 2011.

[6] K. Le, J. Zhang, J. Meng, R. Bianchini, T. D. Nguyen, and Y. Jaluria, "Reducing electricity cost through virtual machine placement in high performance computing clouds," in *SC*, 2011.

[7] I. Goiri, K. Le, M. E. Haque, R. Beauchea, T. D. Nguyen, J. Guitart, J. Torres, and R. Bianchini, "Greenslot: Scheduling energy consumption in green datacenters," in *SC*, 2011.

[8] N. Deng, C. Stewart, and J. Li, "Concentrating renewable energy in grid-tied datacenters," in *ISSST*, 2011.

[9] C. Ren, D. Wang, B. Urgaonkar, and A. Sivasubramaniam, "Carbon-aware energy capacity planning for datacenters," in *MASCOTS*, 2012.

[10] A. Krioukov, C. Goebel, S. Alspaugh, Y. Chen, D. E. Culler, and R. H. Katz, "Integrating renewable energy using data analytics systems: Challenges and opportunities," *IEEE Data Eng. Bull.*

[11] D. Gmach, J. Rolia, C. Bash, Y. Chen, T. Christian, A. Shah, R. Sharma, and Z. Wang, "Capacity planning and power management to exploit sustainable energy," in *CNSM*, 2010.

[12] D. Wang, C. Ren, A. Sivasubramaniam, B. Urgaonkar, and H. Fathy, "Energy storage in datacenters: what, where, and how much?" in *SIGMETRICS*, 2012.

[13] R. Urgaonkar, B. Urgaonkar, M. J. Neely, and A. Sivasubramaniam, "Optimal power cost management using stored energy in data centers," in *SIGMETRICS*, 2011.

[14] R. Singh, D. Irwin, and P. Shenoy, "Yank: Enabling Green Data Centers to Pull the Plug," in *NSDI*, 2013.

[15] K. Le, R. Bianchini, M. Martonosi, and T. D. Nguyen, "Cost- and energy-aware load distribution across data centers," in *HotPower*, 2009.

[16] C. L. Archer and M. Z. Jacobson, "Supplying baseload power and reducing transmission requirements by interconnecting wind farms," *Journal Of Applied Meteorology And Climatology*, 2007.

[17] B. V. Mathiesen and H. Lund, "Comparative analyses of seven technologies to facilitate the integration of fluctuating renewable energy sources," *Journal of Renewable Power Generation, IET*, 2009.

[18] M. A. Delucchi and M. Z. Jacobson, "Providing all global energy with wind, water, and solar power, part II: Reliability, system and transmission costs, and policies," *Energy Policy*, 2010.

[19] D. Chiu, C. Stewart, and B. McManus, "Electric grid balancing through low-cost workload migration," in *GREENMETRICS*, 2013.

[20] L. Lu, J. Tu, C.-K. Chau, M. Chen, and X. Lin, "Online energy generation scheduling for microgrids with intermittent energy sources and co-generation," in *SIGMETRICS*, 2013.

[21] S. Akoush, R. Sohan, A. Rice, A. W. Moore, and A. Hopper, "Free lunch: Exploiting renewable energy for computing," in *HotOS*, 2011.

[22] C. Despins, F. Labeau, R. Labelle, T. Ngoc, J. McNeil, A. Leon-Garcia, M. Cheriet, O. Cherkaoui, Y. Lemieux, M. Lemay, C. Thibeault, and F. Gagnon, "Leveraging green communications for carbon emission reductions: Techniques, testbeds and emerging carbon footprint standards," *IEEE Communications Magazine*, 2010.

[23] S. Akoush, R. Sohan, A. Rice, A. W. Moore, and A. Hopper, "Predicting the performance of virtual machine migration," in *MASCOTS*, 2010.

[24] A. Verma, G. Kumar, R. Koller, and A. Sen, "CosMig: Modeling the impact of reconfiguration in a cloud," in *MASCOTS*, 2011.

[25] S. Akoush, R. Sohan, A. Rice, B. Roman, and A. Hopper, "Activity based sector synchronisation: Efficient transfer of disk-state for wan live migration," in *MASCOTS*, 2011.

[26] J. Zheng, T. S. E. Ng, and K. Sripanidkulchai, "Workload-aware live storage migration for clouds," in *VEE*, 2011.

[27] Y. Chen, S. Alspaugh, D. Borthakur, and R. Katz, "Energy Efficiency for Large-Scale MapReduce Workloads with Signifficant Interactive Analysis," in *Eurosys*, 2012.

[28] Y. Chen, A. Ganapathi, R. Griffith, and R. H. Katz, "The case for evaluating mapreduce performance using workload suites," in *MASCOTS*, 2011.

[29] L. Lu, J. Tu, C.-K. Chau, M. Chen, and X. Lin, "Workload Characterization on a Production Hadoop Cluster: A Case Study on Taobao," in *IISWC*, 2012.

[30] C. L. Abad, N. Roberts, Y. Lu, and R. H. Campbell, "A storage-centric analysis of mapreduce workloads: File popularity, temporal locality and arrival patterns," in *IISWC*, 2012.

[31] A. V. Minwen Ji and J. Wilkes, "Seneca: Remote mirroring done write," in *USENIX*, 2003.

[32] T. Wood, K. K. Ramakrishnan, P. Shenoy, and J. van der Merwe, "Cloudnet: dynamic pooling of cloud resources by live WAN migration of virtual machines," in *VEE*, 2011.

[33] R. Koller, A. Verma, and A. Neogi, "Wattapp: an application aware power meter for shared data centers," in *ICAC*, 2010.

[34] A. Kansal, F. Zhao, J. Liu, N. Kothari, and A. A. Bhattacharya, "Virtual machine power metering and provisioning," in *SoCC*, 2010.

[35] F. Frankovsky, "Data center v1.0 open computer project," http://opencompute.org/, 2011.

[36] "Google Data Center Efficiency," http://www.google.com/about/datacenters/, 2011.

[37] G. Amvrosiadis, A. Oprea, and B. Schroeder, "Practical Scrubbing: Getting to the bad sector, at the right time," in *DSN*, 2012.

[38] J. Hamilton, "Energy proportional datacenter networks," http://perspectives.mvdirona.com/, 2010.

[39] J. Baliga, R. Ayre, K. Hinton, W. V. Sorin, and R. S. Tucker, "Energy consumption in optical IP networks," *Journal of Lightwave Technology*, 2009.

[40] "Cisco CRS-3 16-Slot Single-Shelf System," http://www.cisco.com/, 2010.

[41] "MIDAS Land Surface Stations Data (1853-current), British Atmospheric Data Centre," http://badc.nerc.ac.uk/data/ukmo-midas, UK Meteorological Office, 2006.

[42] D. R. Brooks, "Simple model of solar irradiance," http://www.srrb.noaa.gov/, 2006.

[43] C. Isci, J. Liu, B. Abali, J. Kephart, and J. Kouloheris, "Improving server utilization using fast virtual machine migration," *IBM Journal of Research and Development*, vol. 55, no. 6, pp. 4:1–4:12, 2011.

[44] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in *NSDI*, 2005.

[45] J. Hamilton, "Exploring the limits of datacenter temprature," http://perspectives.mvdirona.com/, 2011.

[46] Google, "Cluster data trace," http://code.google.com/p/googleclusterdata/wiki, 2012.

[47] A. K. Mishra, J. L. Hellerstein, W. Cirne, and C. R. Das, "Towards characterizing cloud backend workloads: insights from Google compute clusters," *SIGMETRICS Perform. Eval. Rev.*, 2010.

[48] P. V. Mieghem, *Performance Analysis of Communications Networks and Systems*. Cambridge University Press, 2005.

[49] P. Gill, M. F. Arlitt, Z. Li, and A. Mahanti, “The flattening internet topology: Natural evolution, unsightly barnacles or contrived collapse?” in *PAM*, 2008.

[50] “400 Gb/s Ethernet Study Group,” http://www.ieee802.org/3/400GSG/, 2013.

[51] A. Sano, T. Kobayashi, E. Yoshida, and Y. Miyamoto, “Ultra-high capacity optical transmission technologies for 100 TB/s optical transport networks,” *IEICE TRANS. COMMUUN.*, 2011.

[52] D. A. Patterson, “Technical perspective: the data center is the computer,” *ACM Commun.*, 2008.

[53] J. Dean and S. Ghemawat, “MapReduce: simplified data processing on large clusters,” *ACM Commun.*, 2008.

A Pseudo-code of the Placement and Scheduling Algorithm

Algorithm 1 Placement and Scheduling Algorithm

```
procedure PLACEMENTALG(start_datetime,end_datetime,dcs,power_trace,computation_trace)
    tick ← start_datetime(computation_trace,tick)
    running_comp_array ← 0
    paused_comp_array ← 0
    delayed_comp_array ← 0
    completed_comp_array ← 0
    while tick < end_datetime do
        delayed_comp_array ← delayed_comp_array ∪ load_comp(computation_trace,tick)
        load_power(power_trace,dcs,tick)
        paused_comp_array ← excess_comp(running_comp_array,dcs)
        for all comp in paused_comp_array do                    ▷ Try to migrated paused apps
            dc_found ← False
            for all dc in dcs do
                if (dc.used_power+comp.power≤dc.current_power) AND (dc.used_machines+comp.machines≤dc.max_machines) then    ▷ Migrate
                    comp.dc_id ← dc_id
                    running_comp_array.add(comp)
                    paused_comp_array.remove(comp)
                    dc.used_power ← dc.used_power + comp.power
                    dc.used_machines ← dc.used_machines + comp.machines
                    comp.run_time ← comp.run_time + tick
                    comp.pause_time ← comp.pause_time + downtime        ▷ Migration downtime
                    dc_found ← True
                    break
                end if
            end for
            if dc_found = False then
                comp.pause_time ← comp.pause_time + tick        ▷ Pausing time due to lack of resources
            end if
        end for
        for all comp in delayed_comp_array do                   ▷ Try to assign new apps
            dc_found ← False
            for all dc in dcs do
                if (dc.used_power+comp.power≤dc.current_power) AND (dc.used_machines+comp.machines≤dc.max_machines) then    ▷ Assign
                    comp.dc_id ← dc_id
                    running_comp_array.add(comp)
                    delayed_comp_array.remove(comp)
                    dc.used_power ← dc.used_power + comp.power
                    dc.used_machines ← dc.used_machines + comp.machines
                    comp.run_time ← comp.run_time + tick
                    dc_found ← True
                    break
                end if
            end for
            if dc_found = False then
                comp.pause_time ← comp.pause_time + tick        ▷ Pausing time due to lack of resources
            end if
        end for
        completed_comp_array ← completed_comp_array ∪ fin_comp(running_comp_array)
        tick ← tick + tick_length
    end while
end procedure
```