Towards Sustainable Cloud Computing

Reducing Electricity Cost and Carbon Footprint for Cloud Data Centers through Geographical and Temporal Shifting of Workloads

by

Trung Le

Thesis submitted to the Faculty of Graduate and Postdoctoral Studies

In partial fulfillment of the requirements for the degree of

Master of Science in Electronic Business Technologies

University of Ottawa

Ottawa, Ontario, Canada

February 2012

Abstract

Cloud Computing presents a novel way for businesses to procure their IT needs. Its elasticity and on-demand provisioning enables a shift from capital expenditures to operating expenses, giving businesses the technological agility they need to respond to an ever-changing marketplace. The rapid adoption of Cloud Computing, however, poses a unique challenge to Cloud providers—their already very large electricity bill and carbon footprint will get larger as they expand; managing both costs is therefore essential to their growth.

This thesis squarely addresses the above challenge. Recognizing the presence of Cloud data centers in multiple locations and the differences in electricity price and emission intensity among these locations and over time, we develop an optimization framework that couples workload distribution with time-varying signals on electricity price and emission intensity for financial and environmental benefits. The framework is comprised of an optimization model, an aggregate cost function, and 6 scheduling heuristics.

To evaluate cost savings, we run simulations with 5 data centers located across North America over a period of 81 days. We use historical data on electricity price, emission intensity, and workload collected from market operators and research data archives. We find that our framework can produce substantial cost savings, especially when workloads are distributed both geographically and temporally—up to 53.35% on electricity cost, or 29.13% on carbon cost, or 51.44% on electricity cost and 13.14% on carbon cost simultaneously.

Acknowledgements

This work would not have been possible without the support of many individuals, to whom I owe my deepest gratitude.

I would like to thank Professor David Wright for his guidance and encouragement throughout this project. Thank you for accompanying me on all those odd tangents I went on before falling into this rabbit hole.

I would like to thank my mother and my sister for their unyielding support over the years. To my beautiful nephew and niece, this is for you.

Many thanks to friends and colleagues for standing by me throughout my studies. You are my source of inspiration.

Table of Contents

1 Introduction

1.1 Problem Statement

Despite the recent economic downturn, Internet use has continued to increase. By 2010, about 30 percent of world population or 2.1 billion people were Internet users (ITU, 2011). That's an addition of 220 million users in one year. To meet growing demand, e-businesses need to expand their computing capacity. Cloud Computing is an attractive option for achieving such expansion.

Among the advantages that set Cloud Computing apart from other computing paradigms is its elasticity. Cloud users have access to a virtually unlimited pool of computing resources including processing, storage, and network transport. They can acquire resources from the pool when the need arises and release them when they are no longer needed. "This elasticity of resources, without paying a premium for large scale, is unprecedented in the history of IT." (Armbrust et al., 2009)

Behind that seemingly limitless amount of computing resources are data centers whose energy use already amounts to millions of dollars in electricity cost and million metric tons of carbon emissions. The annual electricity cost of U.S. data centers is expected to reach $7.4 billion by 2011 (EPA, 2007). Worldwide, spending on enterprise power and cooling is exceeding $30 billion and likely to surpass spending on new server hardware (Scaramella & Eastwood, 2007). And since much of the electricity still comes from fossil-fueled power plants, which are major emitters of greenhouse gases, data center footprint is also enormous, reaching 259 million metric tons of CO_2 equivalent globally by 2020 (The Climate Group, 2008).

The conventional approach to reducing electricity cost and carbon footprint has been to reduce the amount of energy that data centers consume through better energy efficiency. "Free" cooling methods, energy-proportional designs, efficient power conversion, virtualization, and energy-aware resource management have all been proposed as ways to lessen energy demand of data centers. Recent findings suggest, however, that energy efficiency alone is insufficient. At each IT refresh cycle, new servers with better performance will continue to demand more energy due to the fact that gains in efficiency still lag behind gains in performance (Koomey, Belady, Patterson, & Santos, 2009). This means a continual increase in electricity cost and carbon footprint for cloud providers.

But electricity cost and carbon footprint are determined not only by the amount of electricity being consumed but also by electricity price and emission intensity. Recent research work has started to explore a complementary approach: distributing workloads to data centers located in regions where electricity is cheaper or the generation mix of electricity has lower emission intensity. Every

computational job submitted adds an additional workload (and hence some marginal energy consumption) to the data center. By steering job placement from one data center to another, we can geographically shift the marginal energy consumption and take advantage of regional cost differentials. While improving data centers' energy efficiency is a sure way to reduce their energy usage, hence their electricity and carbon costs, routing workloads to data centers in different locations may entail some trade-off between these costs. Shifting workloads to locations powered by coal-fired power plants may be cheaper in financial terms, but that comes with a penalty on carbon footprint as coal-fired power plants emit many times more greenhouse gases than their renewable counterparts (Sovacool, 2008). Our analysis of previous work (see section 2.3.2) finds the trade-off problem between electricity and carbon costs has not been addressed. While previous work such as (Qureshi, Weber, Balakrishnan, Guttag, & Maggs, 2009) and (Garg, Yeo, Anandasivam, & Buyya, 2011) has shown potential savings from skewing user workloads across multiple regions and pushing them into the least expensive regions, the authors focus on either electricity cost or carbon cost and do not address the trade-off between them. High electricity cost hurts cloud providers' bottom line. Hence, reducing this cost is critical for their business. But it is also important for cloud providers to reduce their carbon footprints. Overwhelming scientific evidence on climate change has turned policy makers and customers' attention to the environmental aspect of business. Without sufficient carbon reduction measures, cloud providers can be at risk of violating environmental regulations and losing customers to competitors who offer "greener" products and services.

Another aspect that has been overlooked in previous work is the potential cost savings from temporal shifting of workloads, i.e. delaying workload execution to a later time when electricity is cheaper and/or cleaner. In regions where electricity price changes throughout the day, individual households can save money by postponing energy-intensive activities such as doing laundry to off-peak hours when electricity is cheaper. Such practice can certainly be applied in computing where scheduling is a common task.

Thus, this research work is undertaken in response to two questions: 1) how to address the trade-off between electricity cost and carbon footprint/cost when distributing computational workloads across data centers in different locations and 2) how to achieve cost savings through exploiting temporal and geographical variations in electricity price and emission intensity? The underlying hypothesis is that it is possible to reduce both electricity cost and carbon footprint for Cloud data centers through shifting workloads in time and space.

1.2 Thesis Objectives

We aim to develop a multi-objective optimization framework for workload distribution that is capable of handling the trade-off between electricity and carbon costs and reduces these costs through shifting workloads in time and space. The framework couples workload distribution with electricity price and emission intensity signals and exploits temporal and geographical variations in these signals for financial and environmental benefits. Since it is not always possible to reduce electricity cost and carbon cost simultaneously, the framework provides different optimization heuristics that Cloud providers can use to achieve different trade-offs. To support our theoretical framework, we demonstrate its ability to produce cost savings through simulation using real-world data.

We focus mainly on the initial distribution of user workloads, i.e. when computation jobs are first submitted to cloud providers and routed to selected data centers. Data center selection is based on external factors (e.g., electricity price and emission intensity) and internal factors (e.g., data center energy consumption). These factors are expected to change over time. Electricity prices can go up or down depending on the balance between electricity supply and demand. Emission intensities can fluctuate due to the availability of energy sources. Data centers' energy usage can vary due to changes in existing load or system configuration. To sustain savings over time, long-lived workloads may need to be migrated from one data center to another in response to new conditions. Thus, our work provides an entry point to on-going research on energy-aware workload migration across data centers such as (Buchbinder, Jain, & Menache, 2011).

1.3 Thesis Contributions

This thesis introduces an optimization framework for distributing computational workloads across multiple data centers that squarely addresses the dual energy challenge for Cloud providers, i.e. containing electricity cost while at the same time reducing carbon emissions. The framework comprises three main components—an optimization model, an aggregate cost function, and six workload scheduling heuristics. The model provides a way to gain insights into how cost savings can be achieved by coupling workload distribution with signals on electricity price and emission intensity. The aggregate cost function combines different types of costs such as electricity cost and carbon cost into one figure of merit, facilitating optimization for multiple costs simultaneously. And the heuristics provides different scheduling strategies that Cloud providers can employ to achieve different trade-offs.

The thesis extends the current literature in two ways. Firstly, it tackles the trade-off problem between electricity cost and carbon cost when routing workloads to geographically distributed data centers,

enabling Cloud providers to reduce both costs simultaneously. In comparison, previous work focuses on either electricity cost or carbon cost, but not both. Secondly, it studies the shifting of workloads in both time and space, opening new ways for Cloud providers to cut costs. Previous work only explores shifting workloads in space.

Through simulations driven by real-world data, the thesis also provides insights into how the trade-off between electricity cost and carbon cost would change and how much of these costs could be reduced when shifting computational workloads to different data center locations and times based on signals on electricity price and emission intensity.

1.4 Thesis Organization

The thesis is organized into 6 chapters as follows:

In chapter 1, we present the problem statement, objectives, contributions, and outline of this thesis. We describe the context and starting point of our study, what we want to achieve, its consequences, and how our study is organized.

In chapter 2, we provide background materials for subsequent discussion and explain how our work fits in the current literature. We discuss key concepts and benefits of Cloud Computing and its adoption by businesses. We show trends in energy consumption by Cloud data centers and the challenges that these trends lead to. We describe two main approaches to addressing the challenges—improving data center efficiency and exploiting regional cost differentials—and provide an analysis of previous work exploring the latter approach. Based on the analysis, we identify some of the gaps in the literature that we aim to bridge.

In chapter 3, we describe our overall research design, explaining the steps we take to find answers to our research questions and validate our central hypothesis.

In chapter 4, we present our solution to the research problem: a multi-objective optimization framework for workload distribution that comprises an optimization model, an aggregate cost function, and six optimization heuristics. We formulate an optimization model that couples workload scheduling with electricity price and emission intensity signals. We discuss two common approaches to reconciling competing optimization objectives—sequential optimization and cost aggregation—and propose a weighted sum aggregate function for combining electricity cost and carbon cost into one figure of merit. We describe common optimization algorithms and propose six heuristics for scheduling workloads that

Cloud providers can use to achieve different trade-offs between electricity cost and carbon cost. We conclude the chapter by discussing several important factors that dictate the effectiveness of our framework.

In chapter 5 & 6, we develop energy models and collect real-world data on computational workload, electricity price, and emission intensity; all of which is subsequently used for evaluating our framework through simulation. We build a composite energy model that accounts for IT and non-IT energy use in a data center. We describe a high performance computing workload trace collected at a research lab. We provide an overview of North America' wholesale electricity markets and describe the historical data on electricity price and emission intensity that we gathered from market operators and other sources.

In chapter 7, we evaluate our optimization framework with regards to its design goals. We describe our simulation strategy and present important results from our simulations, which show our framework can provide different trade-off options for Cloud providers and significantly reduces their electricity and carbon costs. We also provide insights into how cost savings responds to changes in job deadline and carbon price.

In chapter 8, we conclude our work by reviewing its contributions, discussing its limitations, and providing suggestions for future research.

2 Background

2.1 Cloud Computing

2.1.1 Concepts & Definitions

Cloud Computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction (Mell & Grance, 2010). Cloud Computing has been identified with five essential characteristics, three service models, and four deployment models.

Essential Characteristics

- **On-demand self-service**. Cloud users can acquire resources from the computing cloud on-demand and release the resources when they are no longer needed. The acquisition and release of resources are performed by cloud users using administration tools provided by cloud providers. Provider involvement is kept to minimum, usually occurring only when there are issues to be resolved.
- **Broad network access**. Resources are made available over the network (typically the Internet) and accessed through standard mechanisms supported by client platforms (e.g., mobile phones, laptops, and PDAs).
- **Resource pooling**. The provider's computing resources are pooled together to serve multiple users using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned to meet user requirements. Cloud users usually have only vague ideas about where the provided resources are located (e.g., country, state, or data center).
- **Elasticity**. Cloud users can quickly scale up or scale down their IT capabilities by purchasing more resources from the cloud or releasing resources back to the cloud. To cloud users, resources from the cloud appear to be unlimited and can be purchased in any quantity at any time.
- **Measured service**. Resources are metered at some level of abstraction appropriate to the type of service (e.g., storage, processing, network bandwidth, and active user accounts). Resource usage is monitored, controlled, and reported to ensure transparency for both the provider and the user.

Service Models

- **Software as a Service (SaaS)**. This model is targeted at the end-user. The capability provided to the end-user is to use the SaaS provider's applications running on a computing cloud. The applications are typically accessible from various client devices through a thin client interface such as a web browser. Examples of SaaS include Salesforce CRM, Gmail, and Microsoft Office 365. The end user does not manage or control the underlying cloud infrastructure or even individual application capabilities, except some basic user-specific application configuration settings.
- **Platform as a Service (PaaS)**. This model is targeted at the application provider (SaaS provider). The capability provided to the application provider is to deploy onto the cloud infrastructure applications created using programming languages and tools supported by the PaaS provider. Examples of PaaS include Google AppEngine, Heroku, and Aptana. The application provider does not manage or control the underlying cloud infrastructure but has control over the deployed applications and possibly application hosting environment configurations.
- **Infrastructure as a Service (IaaS)**. This model is targeted at the application provider (SaaS provider). The capability provided to the application provider is to provision processing, storage, networks, and other fundamental computing resources provided by the IaaS provider. The application provider uses these resources to deploy and run arbitrary software including operating systems and applications. Examples of IaaS include Amazon EC2, Rackspace, and IBM Cloud. The application provider does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly limited control of select networking components.

Deployment Models

- **Private cloud**. The cloud infrastructure is used solely by one organization.
- **Community cloud**. The cloud infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations).
- **Public cloud**. The cloud infrastructure is made available to the general public or a large industry group.
- **Hybrid cloud**. The cloud infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load-balancing between clouds).

Cloud Computing vs. Grid Computing

Cloud Computing is not a completely new concept; it has its root in Grid Computing, Cluster Computing, and distributed systems in general. Compared to Grid Computing, Cloud Computing gives users more agility in provisioning their computing resources. Cloud users can unilaterally acquire resources from the Cloud without any interaction with Cloud providers, whereas Grid users must seek approval from Grid providers. Also, Cloud users can quickly, and in some cases automatically, scale up their Cloud-based infrastructure to meet demand spikes or scale down to reduce expenses. Since resource acquisition in Grid Computing is based on well-defined proposals and Grid users have to wait in queue for their proposals to be approved & processed, such elastic provisioning is not possible.

Computing as the 5th Utility

The ever-increasing importance of computing in daily life routines revitalizes the vision of computing as an essential utility like water, air, electricity, and telephony. Cloud Computing is the latest computing paradigm that promises the delivery of such a vision.

The idea of computing becoming a public utility is not new. In his 1966 book, Douglas Parkhill described a future where many computing activities would be provided through computer utilities, analogous to the electricity industry (Parkhill, 1966). Under this vision, users need to pay providers only when they access computing services. Also, they no longer need to invest heavily or encounter difficulties in building and maintaining complex IT infrastructure. This is like how they get electricity without having to building their own power plants.

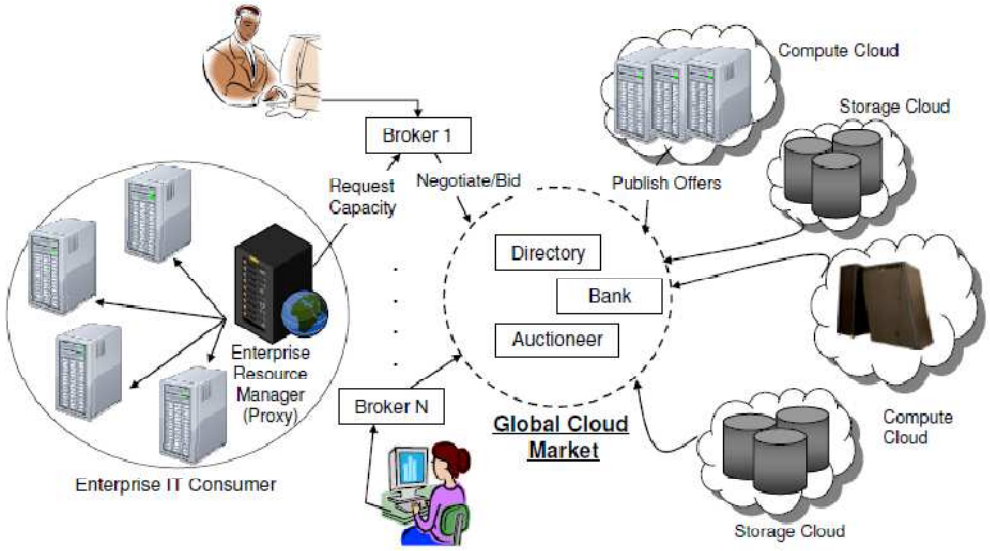


Figure 1: Global Cloud Exchange and Market Infrastructure for Trading Services (Bayya *et la.*, 2009)

Built on previous paradigms such as Cluster Computing and Grid Computing, Cloud Computing promises a virtually unlimited supply of computing resources delivered from data centers to users over a network. Resource usage is charged in a "pay as you go" manner and without the requirement of a long-term contract or upfront commitment. Buyya *et la.* (2009) further articulate that computing under this paradigm can be traded on global cloud exchanges and markets. The market directory allows participants to locate providers or users with the right offers. Auctioneers periodically clear bids and asks submitted by market participants. The banking system ensures financial transactions between participants are carried out properly. Brokers mediate between users and providers by buying capacity from the provider and sub-leasing these to the users. Such markets help bridge disparate clouds, allowing users to choose a provider that suits their requirements by either executing Service Level Agreements in advance or by buying capacity on the spot.

2.1.2 Benefits & Adoption

Benefits that Cloud Computing offers to its users generally amount to cost savings, risk mitigation, complexity reduction, and agility (Armbrust et al., 2009)(Harms & Yamartino, 2010).

Cost Savings. Many companies today are faced with the problem that a large chunk of their IT budget is spent on "keeping the lights on", i.e. operating and maintaining IT hardware and software. Also, their IT infrastructure is often provisioned for peak workload, resulting in waste from the under-utilization of resources at nonpeak times. Cloud Computing helps companies offload these problems to cloud providers whose expertise and massive economics of scale allow them to do a better job at a lower cost. It frees companies from common IT operational headache such as software patching and hardware replacement, enabling them to focus on their core business competencies.

Risk Mitigation. IT capacity planning often involves managing with the risks of over-provisioning and under-provisioning. If companies overestimate peak demands, they waste money on unused resources. If they underestimate peak demands, they risk losing excess users. Even when companies predict their peak demands correctly, they still suffer from non-peak waste. Cloud Computing allows companies to adjust their IT capacity according to the actual demand, thus effectively mitigate these risks.

Complexity Reduction. Managing an IT infrastructure includes coping with the complexity of its hardware and software stacks. Each piece of software requires certain kind of installation, configuration, patching, and integration with other software. Similarly, IT hardware often requires some assembly, tuning, monitoring, and replacement in case of failure. Cloud providers shoulder such heavy lifting and allow users to work with the abstraction layers of their choice. SaaS users, for example, can access business applications through web interfaces without having to install any of the underlying software.

PaaS users can deploy their applications in runtime environments without having to put together all the software and hardware constituting those environments.

Agility. Resource provisioning processes can take weeks or even months in typical IT shops. For companies whose successes depend on how quickly they can respond to customer demand or carry out business operations, the status quo is no longer acceptable. Cloud Computing helps companies overcome this problem by make computing resources available for purchases via on-demand self-services with little or no wait-time.

The future looks bright for the Cloud Computing industry. Market research firm IDC reports that worldwide revenue from public cloud services exceeded $16 billion in 2009 and is forecast to reach $55.5 billion in 2014, representing a compound annual growth rate of 27.4% (IDC, 2010). This is over five times the projected growth for traditional IT products (5%). A recently published study from IBM further confirms this growth prospect (IBM, 2011). The study is based on in-person interviews with 3,018 CIOs, spanning 71 countries and 18 industries. 60% of them plan to integrate Cloud Computing into their technology portfolios to increase business competitiveness. This is a major leap in user interests compared to the result of 33% in 2009.

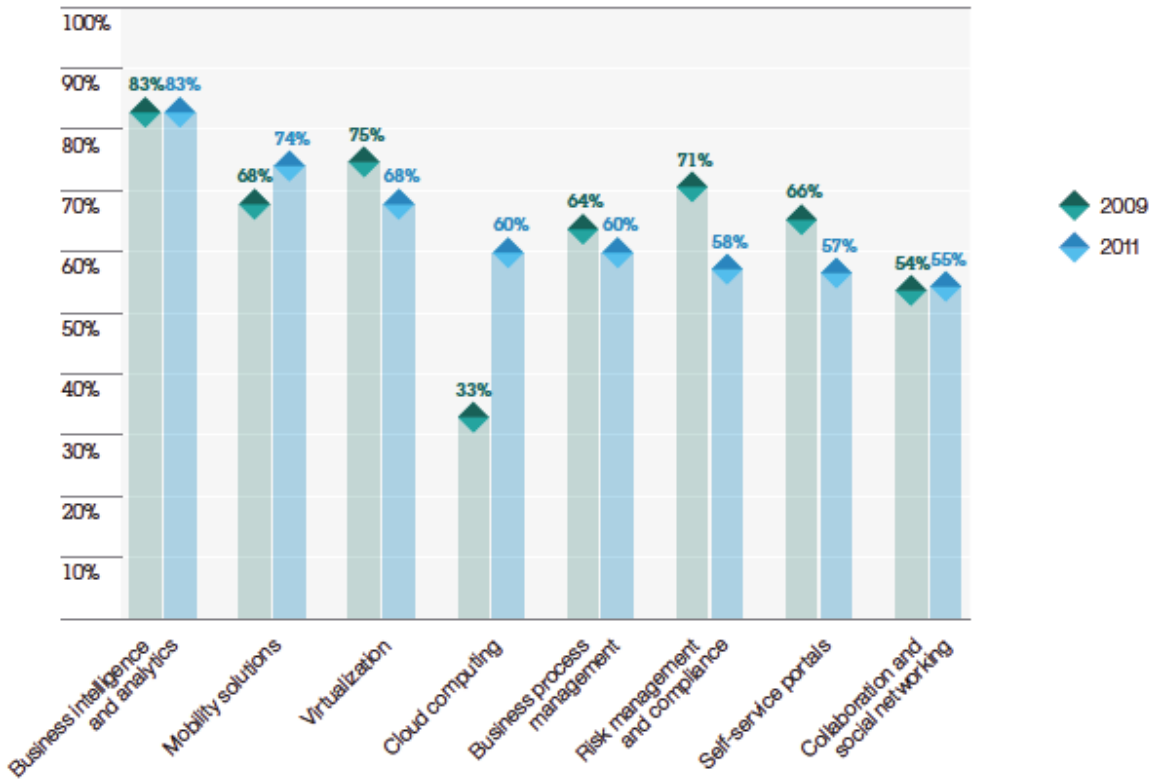


Figure 2: CIO Technology Adoption Vision (IBM, 2011)

2.2 Energy Challenge

While there are great opportunities for the Cloud Computing industry to grow, the opportunities are accompanied with significant challenges. Concerns over security and control of data have lingered from the very beginning and still not been addressed sufficiently. Compliance with increasingly sophisticated law and regulation is difficult to achieve and can be a deal breaker. Availability and performance among other quality-of-service issues demand immediate attention. The lack of standards also raises concerns about vendor lock-in. But perhaps no other challenges have a more direct and extensive impact on the growth of the Cloud Computing industry than its ever-increasing appetite for energy at a time when energy prices are running high and much of the available energy still comes from polluting and non-renewable sources.

2.2.1 Current Trends

Cloud Computing is made possible with data centers packed with thousands of power-hungry servers. Operating these facilities requires an enormous amount of energy. A single data center can consume many megawatts of electricity, as much as tens of thousands of US households. At an aggregated level, U.S. data centers consumed a total of 61 billion KWh in 2006, on par with the amount of energy used by the entire transportation manufacturing sector (including the manufacture of automobiles, aircraft, trucks, and ships) (EPA, 2007). Their peak load on the power grid was estimated to be approximately 7 GW, equivalent to the output of about 15 baseload power plants. If current trends continue, energy consumption by data centers in the U.S. could exceed 100 billion KWh by 2011. The peak load could rise to 12GW and would require an additional 10 power plants.

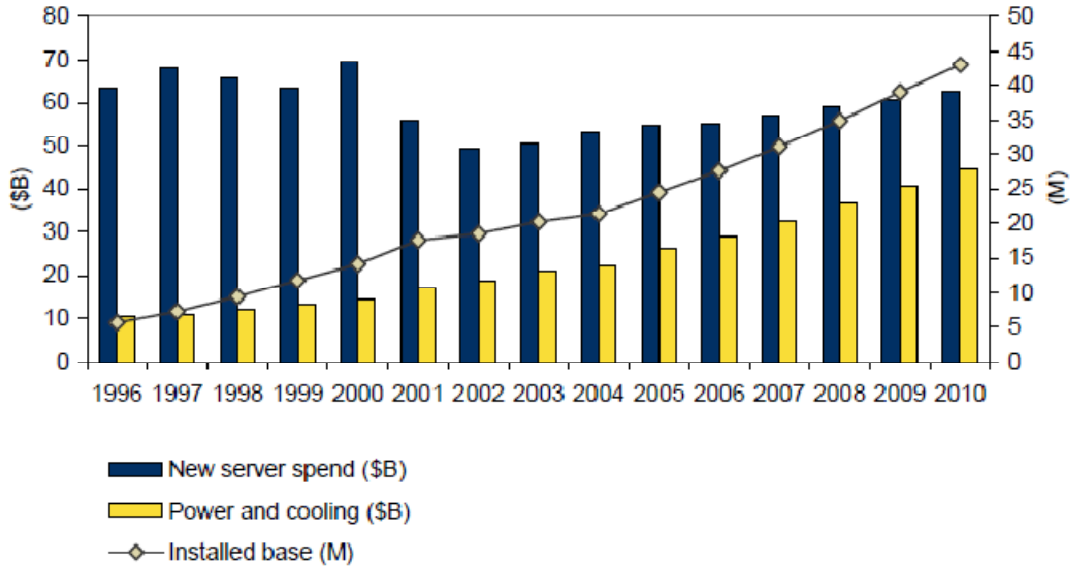


Figure 3: Worldwide Cost to Power and Cool Server Installed Base, 1996-2010 (Scaramella & Eastwood, 2007)

Such scales of energy consumption not only place a huge stress on the power grid but also result in expensive electricity bills for cloud providers. At a wholesale rate of $60 per MWh, Cloud provider Google could face an annual electric bill of $135 million (Qureshi, 2010). The total electricity cost for U.S. data centers in 2006 was about $4.5 billion and could reach $7.4 billion by 2011. Coupled with high energy prices, the cost of powering data center equipment over its useful life is likely to exceed the original capital investment within the next few years if current efficiency trends remain unchanged (Scaramella & Eastwood, 2007).

At the same time, since much of the electricity still comes from fossil-fueled power plants, which are major greenhouse gas emitters, data centers have large and growing carbon footprints. Worldwide data center footprint was estimated at 80 MtCO2e in 2007 (McKinsey, 2008), more than half of the footprint of countries such as Argentina and the Netherlands. Without sufficient reduction measures, this footprint could quadruple by 2020. Carbon emissions are the main contributor to global warming which receives increasing attention from policy makers and customers. Companies that do not reduce their shares of emissions risk violating emerging environmental regulations and losing customers to competitors whose products and services are more environment-friendly.

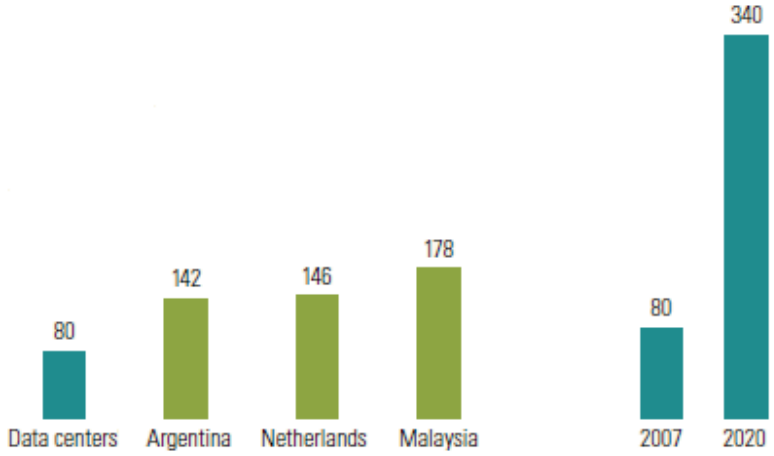


Figure 4: Worldwide Data Center Footprint in megatons of CO2e (McKinsey, 2008)

2.2.2 Sources of Energy Consumption

In a typical raised-floor data center, most energy is spent on IT equipment and cooling systems, or dissipated in conversions within power conditioning equipment (Hoelzle & Barroso, 2009; Pelley, Meisner, Wenisch, & VanGilder, 2009). IT equipment includes servers and network switches. Cooling systems include chillers, computer room air conditioning (CRAC) units, and humidifiers. Power conditioning equipment includes uninterruptable power supply (UPS) units, power distribution units (PDUs), transformers, and switch-gears.

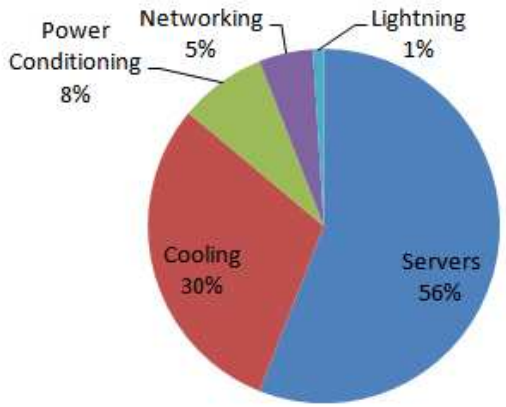


Figure 5: Data Center Power Breakdown (Pelley et al., 2009)

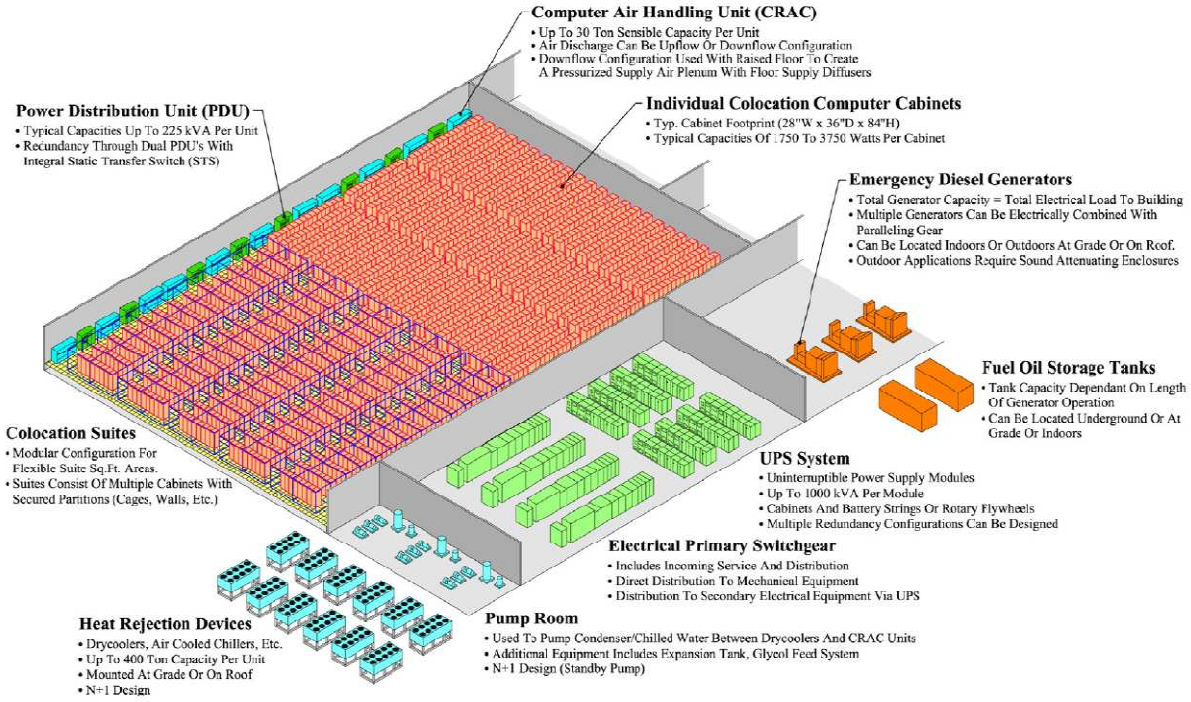


Figure 6: Main Components of a Typical Raised-Floor Data Center (Hoelzle & Barroso, 2009)

Servers. Servers are where energy is translated into useful work. Low-end servers are often the preferred choice for cloud providers primarily because they are more cost-efficient than the high-end shared memory systems that had earlier been the preferred building blocks for the high-performance computing space. They share many key components with the very high-volume personal computing market, and hence benefit substantially from economies of scale.

Today's servers have rated maximum powers (or nameplate values) that are rarely reached in practice due to their often low utilization. An activity profile of a sample of 5000 Google servers over a period of 6 months finds that these servers spend most of their time within the 10-50% CPU utilization range (Hoelzle & Barroso, 2009).

Energy consumed by a server can be further broken down into several portions. CPU still takes the largest portion of server energy, following by DRAM, disks, network cards, and other subsystems.

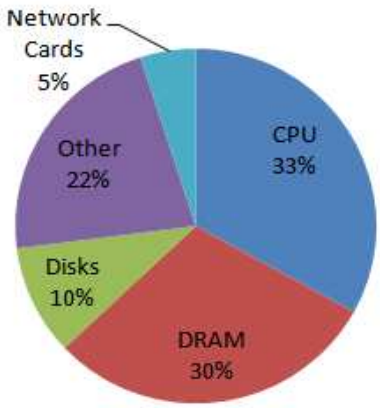


Figure 7: Peak Power Breakdown of a Google Server (Hoelzle & Barroso, 2009)

Network Switches. Network switches tie together clusters of servers. Commodity switches (e.g., 1-Gbps switches with up to 48 ports) are often used to connect servers within a server rack. Switches with high port counts, which are ten times more expensive (per 1-Gbps port) than commodity switches due to their superior bi-section bandwidth, are used to bridge between server racks.

Chillers. Chillers remove heat from liquid coolants. Chillers and other mechanical cooling equipment are usually backed up by generators because the data center cannot operate without cooling for more than a few minutes before overheating. In a typical raised-floor data center, chillers and pumps can add 40% or more to the critical load that needs to be supported by generators.

CRAC. CRAC units pressurize the raised floor plenum by blowing cold air into the plenum. The cold air escapes from the plenum through perforated tiles that are placed in front of server racks, flows through the servers, and expels warm air in the back. Racks are arranged in long aisles that alternate between cold aisles and hot aisles to avoid mixing hot and cold air, which reduce cooling efficiency. CRAC units consist of coils through which a liquid coolant is pumped; fans push air through these coils, thus cooling it. A set of redundant pumps circulate cold coolant to the CRACs and warm coolant back to a chiller, which then expels the heat to the outside environment.

Humidifiers. Humidifiers increase moisture in data centers. According to ASHRAE's recommendations (ASHRAE TC 9.9, 2011), a temperature range of 18 to 27 °C and humidity range of 5.5ºC DP to 60% RH and 15ºC DP are considered optimal for data center conditions.

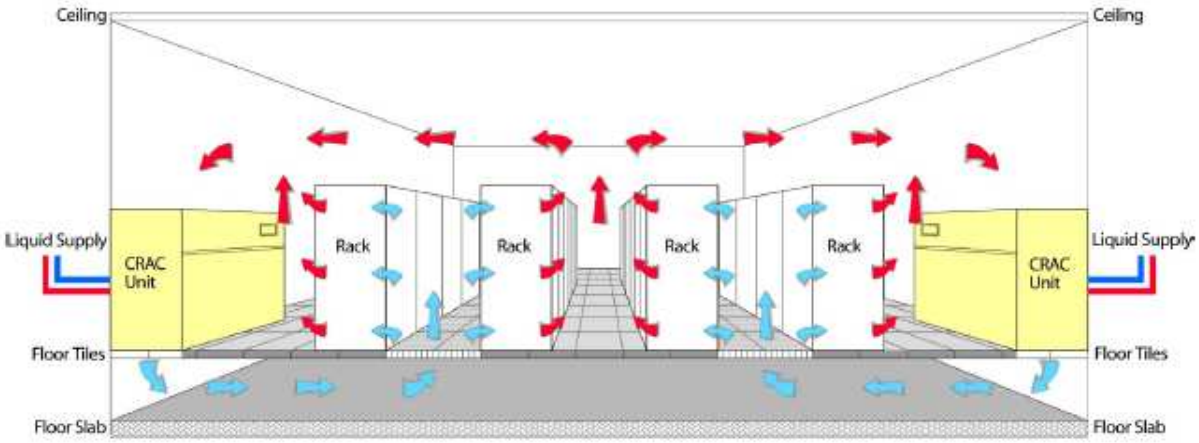


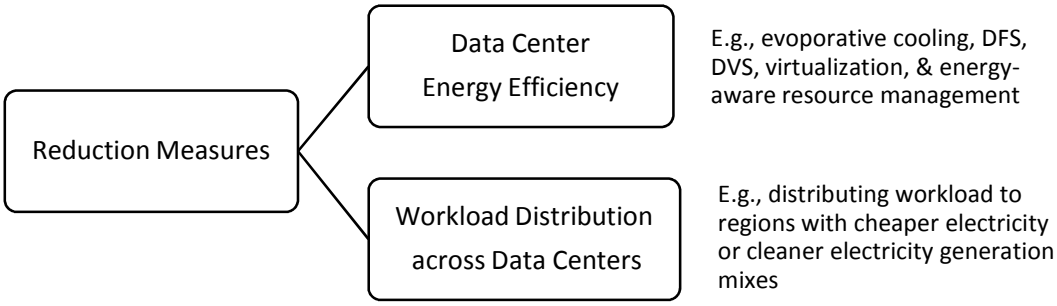
Figure 8: Data center raised floor with hot-cold aisle setup (Hoelzle & Barroso, 2009)

Transformers. Transformers take the medium-voltage power (typically 10-20 kV) from the utility's substation and scale it down to 400-600 V. The low-voltage power then flows into the UPS units. Switch-gears contain breakers to protect IT equipment against electrical faults.

UPS. UPS units have three main functions: (1) switching between utility power and generator power, (2) containing batteries or flywheels to bridge the time between the utility failure and the availability of generator power, and (3) removing voltage spikes or sags, or harmonic distortions in the incoming power feed. Typical UPS sizes range from hundreds of kW up to 2 MW. Output from UPS is routed to power distribution units (PDUs) that sit on the datacenter floor.

PDU. PDUs take a higher-voltage feed (typically 200–480 V) and break it up into the many 110- or 220-V circuits that feed the actual servers on the floor. A typical PDU handles 75–225 kW of load. PDUs often provide additional redundancy by accepting two independent power sources and being able to switch between them with a very small delay so that the loss of one source does not interrupt power to the servers.

2.3 Reduction Measures



2.3.1 Data Center Energy Efficiency

The conventional approach to reducing electricity costs and carbon emissions associated with data center operations has been to reduce the amount of electricity that data centers consume through energy efficiency improvements.

The energy efficiency of a data center is broadly defined as the amount of computational work divided by the total energy used in the process (Hoelzle & Barroso, 2009). Green Grid proposes the Datacenter Performance Efficiency (DCPE) metric to capture this notion (Green Grid, 2007). The general idea is to run a standardized datacenter-wide workload such as SPEC or TPC benchmark and measure the total energy consumed. Barroso & Hölzle (2009) argue that DCPE is not a practical metric because it is very hard to measure—few institutions are able to measure it because the servers running in their datacenters are running actual applications and thus are not available for benchmarking. Instead, they propose DCPE to be factored into three separate components: a facility term (a), a server energy conversion term (b), and the efficiency of the electronic components in performing the computation per se (c). These components are then independently measured and optimized by the appropriate engineering disciplines.

$$\text{Efficiency} = \frac{\text{Computation}}{\text{Total Energy}} = \underbrace{\left(\frac{1}{\text{PUE}}\right)}_{\text{(a)}} \times \underbrace{\left(\frac{1}{\text{SPUE}}\right)}_{\text{(b)}} \times \underbrace{\left(\frac{\text{Computation}}{\text{Total Energy to Electronic Components}}\right)}_{\text{(c)}}$$

- The first component in the efficiency calculation (a) is the power usage effectiveness (PUE), which reflects the quality of the datacenter building infrastructure. PUE captures the ratio of total building power to IT power, that is, the power consumed by the actual IT equipment

(servers, network equipment, etc.). A PUE of 2.0 means for each watt consumed by IT equipment, another watt is used for cooling and overhead.

- The second component (b) in the efficiency calculation is server PUE (SPUE), which accounts for power used in the server's power supply, voltage regulator modules (VRMs), and cooling fans. It consists of the ratio of total server input power to its useful power, where useful power includes only the power consumed by the electronic components directly involved in the computation (such as CPU, disks, DRAM, and network cards).
- The third component (c) in the efficiency calculation accounts for how much of the energy delivered to electronic components in a system is actually translated into useful work.

Large volume of research work has been done in the area of data center energy efficiency. We describe below a few notable techniques for improving the efficiency of various components of the data center. Comprehensive review of the literature on this topic can be found in (Beloglazov, Buyya, Lee, & Zomaya, 2011) and (Berl et al., 2010).

Evaporation. Evaporation is a powerful tool to reduce energy spent on cooling. In a data center cooling system, when chillers are required to run, they can consume many times more energy than the rest of the cooling system combined. Evaporative cooling methods minimize the time chillers need to run by employing cooling towers. Hot water from the data center is released from the top of the cooling tower. As the water flows down, some of the water evaporates, drawing energy out of the remaining water and causing it to cool down.

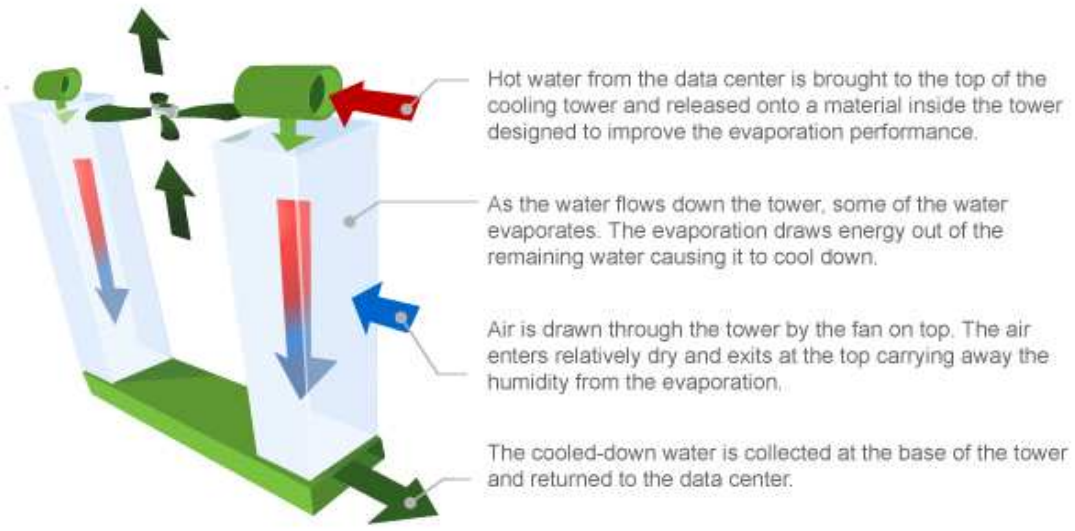


Figure 9: Evaporative cooling (Google, 2011)

Dynamic Frequency and Voltage Scaling. Dynamic frequency scaling (DFS) and dynamic voltage scaling (DVS) are two well-known techniques for conserving energy and reducing heat generated by server CPU. When a server is under-loaded, DFS can be used to lower its CPU frequency (i.e. the number of instructions a CPU can issue in a given amount of time), thus reducing the CPU's dynamic power and heat output. Intel's SpeedStep and AMD' Cool'n'Quiet are two examples of DFS. DVS works on the same principles as DFS but instead of CPU frequency it is the voltage that is dynamically adjusted.

Virtualization. Virtualization is the primary method for consolidating multiple under-utilized servers into one server, hence reducing idle waste. Many applications often need only a small fraction of the available resources of a server. However, even at idle, servers still draw about 60% of their peak power (Barroso & Holzle, 2007). This is due to the fact that main server components have narrow dynamic power ranges: 70% of peak power for CPU, less than 50% for DRAM, and 25% for disk drives. Energy savings can be achieved by running applications within virtual machines consolidated on a smaller number of physical servers, allowing unused servers to be turned off. Since less hardware is deployed, less energy is needed for powering and cooling the data center.

Energy-aware Resource Management. Energy-aware resource management techniques such as live virtual machine migration can be applied at the data center level to exploit other savings opportunities. Under energy-aware schemes, virtual machines are first placed in physical nodes in such a way that minimize the marginal energy consumption. This is usually achieved by provisioning virtual machines using the smallest number of servers. Over time, however, the initial placement may no longer be optimal due to changes in server load or configuration. Migration solutions such as pMapper (Verma, Ahuja, & Neogi, 2008) and GreenCloud (Liu et al., 2009) can be employed to move virtual machines from one server to another so that the overall energy footprint is kept minimal.

While significant reduction in energy use has been achieved through energy efficiency endeavors, the data center footprint continues to rise. This can be attributed to the phenomenal growth in Internet user population and the proliferation of data-intensive computing services such as financial modeling and analysis, video streaming, and social networking. It can also be attributed to the gap between efficiency gains and performance gains.

Brill (2007) observes that between 2000 and 2006 server compute performance increased by a factor of 27 while its energy efficiency only went up by a factor of 8. Koomey *et la.* (2009) confirms this trend through their assessment of server performance, purchase costs, and energy use over time. They examine several families of servers and find performance per dollar of server cost has in all cases increased more rapidly than performance per watt in recent years. As a result, the indirect costs for

cooling and power distribution, which are directly related to the power use per dollar of server acquisition cost, start to offset the performance related benefits of Moore's law. Thus, Cloud providers who do not assess the total cost for purchasing new servers but instead focus solely on performance per dollar of server acquisition cost will invariably overestimate the benefits from buying more computing power.

2.3.2 Workload Distribution across Data Centers

While energy efficiency aims to achieve more computational work with less energy use, workload distribution produces savings by exploiting regional differences in external factors such as electricity price and emission intensity. The key idea is to distribute workloads to data centers located in regions where electricity is cheaper or the generation mix of electricity has lower emission intensity. Every job submission adds an additional workload to the data center, inducing some marginal energy consumption. By redirecting job submissions from one data center location to another, we can shift the marginal energy consumption in space and take advantage of regional cost differentials.

Table 1 gives a comparison of previous work with regards to three aspects—addressing trade-off between electricity and carbon costs, exploiting geographical variations, and exploiting temporal variations in electricity price and emission intensity. To the best of our knowledge, none of the previous work has collectively studied all of these aspects.

Addressing trade-off. Shifting workloads from one data center location to another may entail some trade-off between electricity cost and carbon cost, especially when the electricity prices and emission intensities of those locations are not correlated. Having control over this trade-off allows Cloud providers to cater to different customers who have unique financial and environmental constraints.

Exploiting geographical variations. The electricity price and emission intensity often vary from one region to another, depending on the unique supply-demand balance and generation mix of each region. Cloud providers can cut costs by shifting workloads to cheaper and/or greener regions, or in other words, by following cheap/green power.

Exploiting temporal variations. As the supply-demand balance and generation mix change over time, so do the electricity price and emission intensity. Cloud providers can reduce costs by postponing workload execution to later times when the condition is more favorable, or in other words, by waiting for cheap/green electricity.

Table 1: Comparison of Related Work			
	Addressing Trade-Off	Exploiting Geographical Variations	Exploiting Temporal Variations
Our work	Yes	Yes	Yes
(Qureshi et al., 2009)	No	Yes	No
(Kien Le, Bianchini, Nguyen, Bilgir, & Martonosi, 2010)	No	Yes	No
(Rao, Liu, Xie, & Liu, 2010)	No	Yes	No
(Sankaranarayanan, Sharangi, & Fedorova, 2011)	No	Yes	No
(Garg et al., 2011)	No	Yes	No

An early work in this research direction is (Qureshi et al., 2009). The authors study the problem of dynamic request distribution based on hourly electricity prices. They observe that in regions with wholesale electricity markets, prices vary on an hourly basis and are often not well correlated at different locations. Also, large distributed systems already incorporate request routing and replication to provide good performance and tolerate faults. Based on these observations, they argue that existing distributed systems should be able to exploit the variation in electricity prices for economic gains. Their goal is to provide a method for mapping client requests to geographically distributed clusters so that the overall electricity cost of such a system (in dollars, not Joules) is minimized. Their main contribution is to identify the relevance of electricity price differentials to large distributed systems and provide an estimate on the potential cost savings.

To show that electricity prices exhibit imperfectly correlated variations, the authors analyze 39 months of hourly electricity prices from 29 U.S. locations. The historical data is gathered from government sources, trade publication, and different regional transmission organizations. They find that the

electricity price at each location exhibit a significant amount of day-to-day volatility, short-term spikes, seasonal trends, and dependencies on fuel prices and consumer demand. During a single day, the price per MWh changed hourly by $20 or more roughly 20% of the time. The minimum and maximum prices can differ by a factor of 2. More importantly, they find that locations in different regional markets are not well correlated, even when nearby. The price differential between two locations often vary in time and has a differential distribution with a near-zero mean and a high variance. In terms of differential duration, short differentials (less than 3 hours) are most frequent, medium length differentials (less than 9 hours) are common, and differentials that last longer than a day are rare.

These findings are significant because they provide quantitative evidence supporting the underlying hypothesis that there are savings opportunities due to price variations. Shifting workloads to different locations will not reduce electricity costs if hourly prices at those locations are well correlated.

To project savings, the authors conducted a number of discrete-time simulations, quantifying and analyzing the impact of different request routing policies on energy costs and client-server distance. They evaluate two routing schemes: one based on the original request traces and the other optimized on electricity prices (i.e. mapping requests to clusters with the lowest electricity prices). Real-world hourly electricity prices are obtained from a number of data sources. The request traces come from a large content distribution network. The energy model is adapted from an empirical study on a Cloud data center, in which server power usage is derived from CPU utilization. They find that electricity costs can be reduced by up to 40%. For large distributed systems, this can translate to multi-million dollar savings per year.

This work can be improved in several ways. Firstly, their simulations use web traffic traces which contain a large fraction of computationally trivial hits. For Cloud providers such as Amazon, the types of workload they need to handle can also be computationally intensive. Users may purchase virtual machines from Amazon to perform CPU-demanding tasks such as scientific research, financial modeling and analysis, or video encoding. Such workload has a different energy use profile and hence requires further investigation. Secondly, the authors use a rather simple request routing scheme; user requests are simply mapped to clusters with the lowest electricity prices. Better results could be obtained with routing schemes that employ optimization algorithms. Lastly, the authors focus solely on electricity prices and ignore their environmental implication. As we discuss earlier, the cheapest electricity is often not the cleanest. Without conscious consideration, financial gains may be made at the expense of environmental costs.

A similar problem is studied in (Kien Le et al., 2010) but the focus of this work shifts from exploiting price variations to capping brown energy consumed by Internet services. Brown energy refers to the energy produced via carbon-intensive means such as burning coal. The authors argue that while reducing energy costs is important, it is also necessary for Internet data centers to manage their usage of brown energy given the emergence of Kyoto-style carbon caps. The caps may be government-mandated, utility-imposed, or voluntary. Their central research question is how to create software support for capping brown energy consumption without excessively increasing costs or degrading performance.
The authors address their research question in the context of Internet services that are supported by multiple data centers. The data centers sit behind front-end devices that inspect each client request and forward it to one of the data centers that can serve it, according to a request distribution policy. The authors claim the following contributions: (1) a general, optimization-based framework for minimizing the energy cost of services in the presence of brown energy caps; (2) an optimization-based request distribution policy for minimizing the energy cost while abiding by SLAs; (3) a heuristic policy for the same purpose; and (4) extensive evaluation of the proposed framework and distribution policies through simulation and experimentation.
Their proposed software framework comprises a list of parameters; key parameters include the power mix for each data center and the fraction of requests that should be sent to each data center during an epoch. The power mixes are contracted to the corresponding power utility for an entire year. The request fractions are recomputed after each epoch. The brown energy cap for an Internet service is associated with its data centers and is specified in KWh. When a service exceeds its cap, it must purchase carbon offsets on the market. The brown energy cap for each data center is derived from its predicted load.
Based on this framework, the authors propose an optimization-based policy and a heuristic policy for request distribution. The policies differ in how the request fractions are determined. The optimization-based policy uses Simulated Annealing to solve the optimization problem. The heuristic policy is greedy; it tries to forward each request to the best data center that can serve it (based on a specific metric) without violating the load capacity and completion deadline constraints.
To evaluate their frameworks and policies, the authors use both simulation and real-system experimentation. The simulator includes a single front-end located on the East Coast of the US, distributing requests to 3 data centers located on the West Coast, the East Coast, and in Europe. The simulator is validated against a real prototype implementation running on servers located in the same regions. The authors use a 1-month-long request trace from a commercial search engine, on-peak/off-

peak electricity prices, and carbon market traces from a market intelligence provider. Each request is assumed to take 400ms to process and consumes 60J of dynamic energy plus base energy. They compare three policies: optimization-based, cost-aware heuristic, and cost-unaware heuristic. They find that optimization-based policy achieves up to 35% lower costs than cost-aware policy and 39% lower costs than cost-unaware policy. They also find that lowering the brown energy cap from 100% to 75% enables a savings of 24% in brown energy consumption at only a 10% increase in cost, although a further 50% cap decrease yields a savings of 30% at a much higher cost increase (24%). These results suggest that services can significantly reduce their brown energy consumption at modest cost increases as long as the caps are set appropriately.

This work extends (Qureshi et al., 2009) in that it takes into account the environment aspect of data center energy consumption. Proposed request distribution policies are designed to reduce costs without exceeding caps on brown energy consumption. Caps are contracted to power utilities on a yearly basis and it is up to the utilities to enforce the green/brown energy ratios.

While this strategy might be easy for Cloud providers to implement, it neither encourages nor enables them to aggressively pursue carbon reduction. The premise of carbon capping is that it treats carbon footprint as a constraint rather than an optimization objective. Thus, Cloud providers only need to keep their footprints within the limit, but there is no incentive for them to do better. In addition, using contracts as the only interface with power utilities means Cloud providers are not exposed to important information such as when and where green (renewable) energy is available. Just as energy costs can be reducing by shifting workloads to less expensive regions (i.e. following cheap energy), carbon footprints can be reduced by shifting workloads to less carbon intensive regions (i.e. following clean energy).

Rao *et at.* (2010) study the problem of minimizing the total electricity cost in multi-region electricity markets while guaranteeing quality of service. They point out that service providers build their data centers in different regions; some of which have wholesale electricity markets with prices varying on an hourly basis while others still have regulated markets with prices remaining unchanged for long periods of time. Based on that observation, the authors argue that the exploitation of price fluctuations must be considered in a multi-region electricity market. Their main contributions include (1) a mixed-integer optimization formula for minimizing the total electricity cost of Internet data centers in multi-region electricity markets (2) a polynomial-time approximation method.

The authors formulate the total electricity cost minimization problem as a constrained mixed-integer programming problem. Since client workloads and electricity prices are time-dependent and may change fast, the optimization problem must be solved quickly. The authors, therefore, approximate the

problem using a linear programming formulation, and then convert the formulation to a minimum cost flow problem which can be solved with a polynomial-time algorithm.

For performance evaluation, the authors compare the optimal workload assignment (obtained through solving the optimization problem) and the average workload assignment. They use hourly electricity prices at three locations, two in the deregulated electricity market regions and one in the regulated market region with fixed electricity rate. The price data corresponds to one day in May 2009. Servers are assumed to be operating at 120 Watts. Requests are coming in at a constant rate. They find that their proposed algorithm can reduce the total electricity cost up to 30.15%.

While the authors raise a good point on the need for fast optimization solvers, their performance evaluation does not provide any data to show how efficient their approximation method is. Also, it is unclear how the average workload assignment in their comparison is calculated. Further improvement can be made by expanding the dataset on electricity prices and using more sophisticated energy models.

S.K. Garg *et al.* (2011) investigate the problem of energy and carbon efficient scheduling of high performance computing (HPC) applications in geographically distributed Cloud data centers. They argue that Cloud providers can achieve optimal energy sustainability of running high performance computing (HPC) workloads across their Cloud infrastructure by harnessing the heterogeneity of multiple data centers in different locations worldwide. Their goal is to outline how managing resource allocation across multiple locations can have an impact on the energy cost and carbon footprint of a Cloud provider. The authors claim the following contributions: (1) a novel mathematical model of energy efficiency based on various contributing factors, 2) near-optimal energy-efficient scheduling policies that minimize carbon emissions and maximize profits, (3) analysis of proposed policies through extensive simulation using real HPC workload traces, (4) analysis of lower/upper bounds of the optimization problem, and (5) exploitation of local minima in Dynamic Voltage Scaling to further reduce energy consumption.

The research problem is examined in a Cloud Computing environment where Cloud users are able to tap the computational power offered by Cloud providers to execute their HPC workloads. Cloud users submit their computation jobs to a Cloud meta-scheduler who interprets and analyzes the service requirements and decides whether to accept or reject the job based on the available capacity.

The authors develop an energy efficiency model that includes various factors such as energy cost, emission intensity, HPC workload, and CPU power efficiency. They propose five scheduling policies: two of which minimize carbon emissions, two maximize the profit of resource providers, and the last one is a multi-objective policy that minimizes carbon emissions and maximizes the profit. The multi-objective

policy finds for each application a data center, which provides the least carbon emissions, among data centers able to complete a workload by its deadline. Then among all the application-data center pairs, the policy chooses one, which results in the maximal profit. These steps are repeated until all the workloads are scheduled. They provide a lower bound for the carbon emissions and an upper bound for the profit, both based on the principle that the minimum carbon emission and the maximum profit are achieved when most of the workloads is executed at the most "efficient" data center and also at the optimal CPU frequency. The authors propose the use of DVFS for all the CPUs in data centers to further reduce energy consumption. The CPU frequencies are adjusted so that their energy consumption can be minimized without violating application deadlines. To identify the local minima where the energy consumption will be the minimum, they differentiate the energy consumption of a workload on a CPU at a data center with respect to the operating CPU frequency.

The authors evaluate their scheduling policies through simulation. They use a week-long workload trace extracted from a public parallel workload archive and annual averages of emission intensities and electricity prices published by the U.S. government. Server energy usage is estimated based on CPU frequency. The power usage efficiency of data centers is randomly generated using a uniform distribution. The policies are compared with each other for different scenarios and also with the derived The proposed heuristics are evaluated using simulation of different scenarios. The simulation results show that the energy-centric policies allow the reduction of energy costs by 33% on average.

One aspect that has been overlooked in this work is the granularity of their data on electricity price and emission intensity. Unlike (Qureshi et al., 2009) which uses hourly electricity prices, the authors use annual averages. They make the same choice of data granularity for emission intensities. While annual average values are good indicators, they hide all the dynamics that can only be observed at a finer granularity, such as the fluctuation of electricity prices and emission intensities from one hour to another. Distributing current workloads based on annual indicators may not capture savings opportunities from such short-term fluctuations; or worse yet, it could introduce additional penalties.

Further improvement can also be made in the way they handle competing optimization objectives. While optimizing for one objective after another is a possible way to achieve reconciliation, it has an inherent weakness: objectives at the end of the optimization pipeline may never be considered. The implication is that Cloud providers may not be able to achieve the compromises they want.

Sankaranarayanan *et la.* (2011) study the problem of dispatching service requests across heterogeneous data centers located in different geographical regions. They argue that since data centers are becoming heterogeneous due to hardware upgrade over time and they are often located in different electricity

markets, both of these factors should be exploited for cost reduction purposes. Their goal is to develop a request dispatch solution to minimize data center electricity cost while at the same time keeping the application response time acceptable. The authors claim that their work is the first that considers a combined strategy of exploiting both the heterogeneity of data center server hardware and the energy market price variability towards minimizing the electricity cost.

The authors attempt to minimize the electricity cost at the global level by exploiting the energy price variations in different markets and at the data center level by exploiting the heterogeity of server hardware. At the global level, they propose three load balancing strategies: one evenly distributes service requests across available data centers, one skews requests towards data centers with the cheapest electricity, and one skews requests towards data centers with the smallest latency. At the data center level, they propose two scheduling strategies: one assigning requests to servers in proportion to their service rates and the other assigning requests to the fastest servers first. Inactive servers are turned off.

To evaluate the proposed strategies, the authors use simulation based on real-world workload traces and electricity prices. The workload traces are extracted from Wikipedia and account for 89% of the total requests handled by the site. Hourly electricity prices for an entire week are obtained from two countries in two different time zones. Server power consumption is estimated based on its active and idle profiles. Network delay is based on the ping latency. They find that their strategies outperform existing solutions that do not exploit the electricity market diversity or do not exploit data center hardware diversity.

This work extends the literature in that it takes into account the heterogeneity of data center hardware. Like most of the previous work, however, it does not address the problem with carbon emissions associated with data center energy consumption. Also, its use of national averages can be misleading. While the time granularity of electricity price is per-hour, the space granularity is per-country. National averages can be very different from regional averages. Even for regions that are close to each other, their electricity prices may not be well correlated, as found in (Qureshi et al., 2009).

3 Research Design

As we seek new insights into the financial and environmental benefits of distributing workloads across multiple data centers, we undertake an experimental simulation study. We develop our models, collect input data, and experiment with the models by way of simulation. The results from the simulation allow us to answer our research questions and ultimately validate our central thesis. The structure of our investigation can be illustrated as follows:

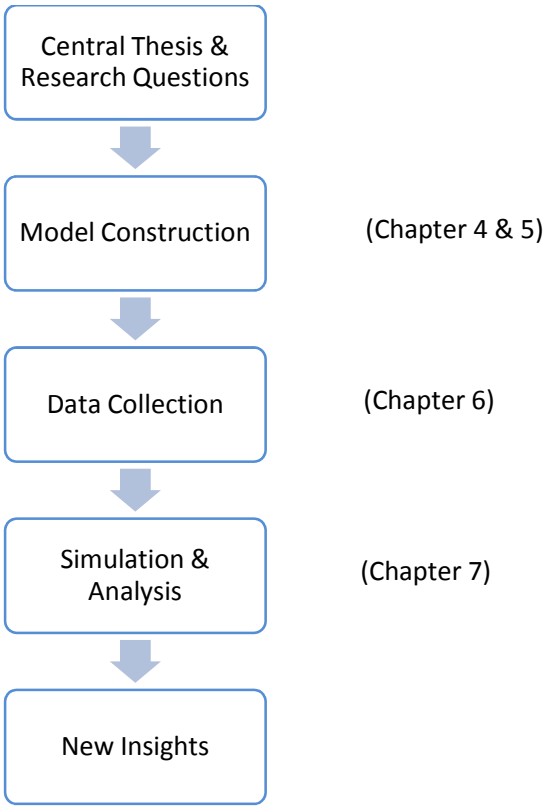


Figure 10: Structure of Investigation

Model Construction. Modeling allows us to study certain aspects of a complex system whose actual construction is cost-prohibited (Borshchev & Filippov, 2004). In this study we seek to understand how the energy-related costs of a computing cloud respond to external signals, i.e. electricity price and emission intensity. An important outcome from model construction is the requirements for input data, which are used in simulate model behaviors. We describe our models in chapter 4 and 5.

Data Collection. To produce realistic estimates of the potential savings from exploiting geographical and temporal differentials, we endeavor to collect real-world data. Running our models requires 3 sets of data—workload, electricity price, and emission intensity. The data on workload is extracted from a real-world trace recorded at a high-performance computing research lab. The data on electricity price and emission intensity are collected from public data archives of wholesale electricity market operators in Canada and the States. We describe our data collection in chapter 6.

Simulation. Simulation is the process of model execution, i.e. taking the model through discrete or continuous state changes over time (Borshchev & Filippov, 2004). In our context, it is the process of feeding data on workload, electricity price, and emission intensity to our energy-related cost model for a computing cloud over a period of time.

The major approaches in simulation modeling are System Dynamics, Discrete Event, and Agent Based. A comparison of these three approaches can be found in (Borshchev & Filippov, 2004). System Dynamics deals with continuous processes whereas Discrete Event and Agent Based work mostly with discrete processes.

We chose Discrete Event simulation over the other approaches since it fits best with our optimization model and input data. System Dynamics is not a good fit because our input data are of discrete nature, e.g., hourly electricity price & hourly emission intensity. Agent Based is not a good fit either as it drives model behaviors through interaction between multiple actors whereas workload distribution in our optimization model is performed by a central scheduler. It is worth noting that previous works in this research area such as (Qureshi et al., 2009) and (Rao et al., 2010) also employ Discrete Event simulation.

We run our optimization model through a discrete event process. The optimization is performed by a central Cloud scheduler and simulated on an hourly basis. For each hour, the Cloud scheduler receives a list of jobs from a workload generator and uses price and emission intensity signals from individual data centers to decide where and when the jobs will be run. This type of simulation is acceptable for workloads that can tolerate some delay before its execution starts. An example is batch jobs that involve intense computation but none of user interaction; what is more important here is when the batch jobs finish rather than when they start. Thus, instead of executing jobs as they arrive, the Cloud provider can postpone their servicing to a later time when electricity is cheaper and/or greener and then use more computing capacity to compensate for the delay. Under the Cloud Computing paradigm, using 1000 servers for one hour is the same as using one server for 1000 hours. Thus, a batch job can be completed

as quickly as it can scale out (e.g., broken up into smaller pieces that can be executed in parallel by multiple servers).

In fact, Infrastructure-as-a-Service providers such as Amazon offers a spot pricing model in which users bid on spare computing capacity and own this capacity for as long as their bid exceeds the current spot price. The presence of such a model indicates that users are willing to accept some flexibility regarding when their workload is executed in exchange for some cost savings.

More details about our simulation strategy are given in chapter 7.

4 Multi-Objective Optimization Framework

In this chapter we propose a multi-objective optimization framework that i) addresses the trade-off between electricity and carbon costs and ii) takes advantage of geographical and temporal cost differentials. Our first goal in designing the framework is to provide different trade-off options for Cloud providers. Our second goal is to achieve cost savings through exploiting geographical and temporal variations in electricity price and emission intensity. Our framework is comprised of three components—a multi-objective optimization model (see section 4.2), an aggregate cost function (see section 4.3.3), and optimization heuristics (see section 4.4.3).

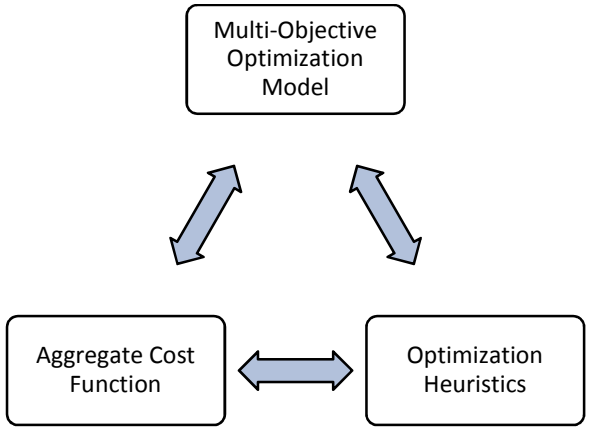


Figure 11: Multi-Objective Optimization Framework for Workload Distribution across Data Centers

4.1 Study Context

We position our study in the context of a public Cloud environment where Cloud users are able to purchase computing capacity from a Cloud provider to execute their workloads. The distribution of user workloads to data centers in the Cloud is done through a central Cloud scheduler, which is managed by the Cloud provider (see Figure 12).

Cloud User. Cloud users need to run various computing workloads. Users submit their workload requirements (e.g., the number of CPU hours and completion deadlines) to the Cloud scheduler. Optionally they can specify their wishes for optimization objectives (e.g., optimizing for financial cost, environmental cost, or any combination of these two).

Cloud Provider. The Cloud provider has multiple data centers that are geographically distributed. Data centers have different capacities, ranging from dozens of servers to thousands of servers. Servers are

homogeneous within a data center but can be different across data centers. The financial cost that users have to pay for executing their workloads at a data center is assumed to be proportional to the electricity cost incurred on that data center at that point in time. A cloud provider may choose to price its services differently, e.g., in order to achieve roughly constant server utilization. In that case, our model can be run using the specified price schedule. The environmental cost or carbon cost is derived from the emission intensity of the generation mix that supplies electricity to the data center at that point in time. The dollar and environmental costs can vary with time of day, day of week, and time of year.

Cloud Scheduler. The Cloud scheduler is at the heart of the Cloud infrastructure. It receives workload/job submission from Cloud users and status updates from data centers, computes the optimal workload distribution in both space and time, and distributes workloads accordingly.

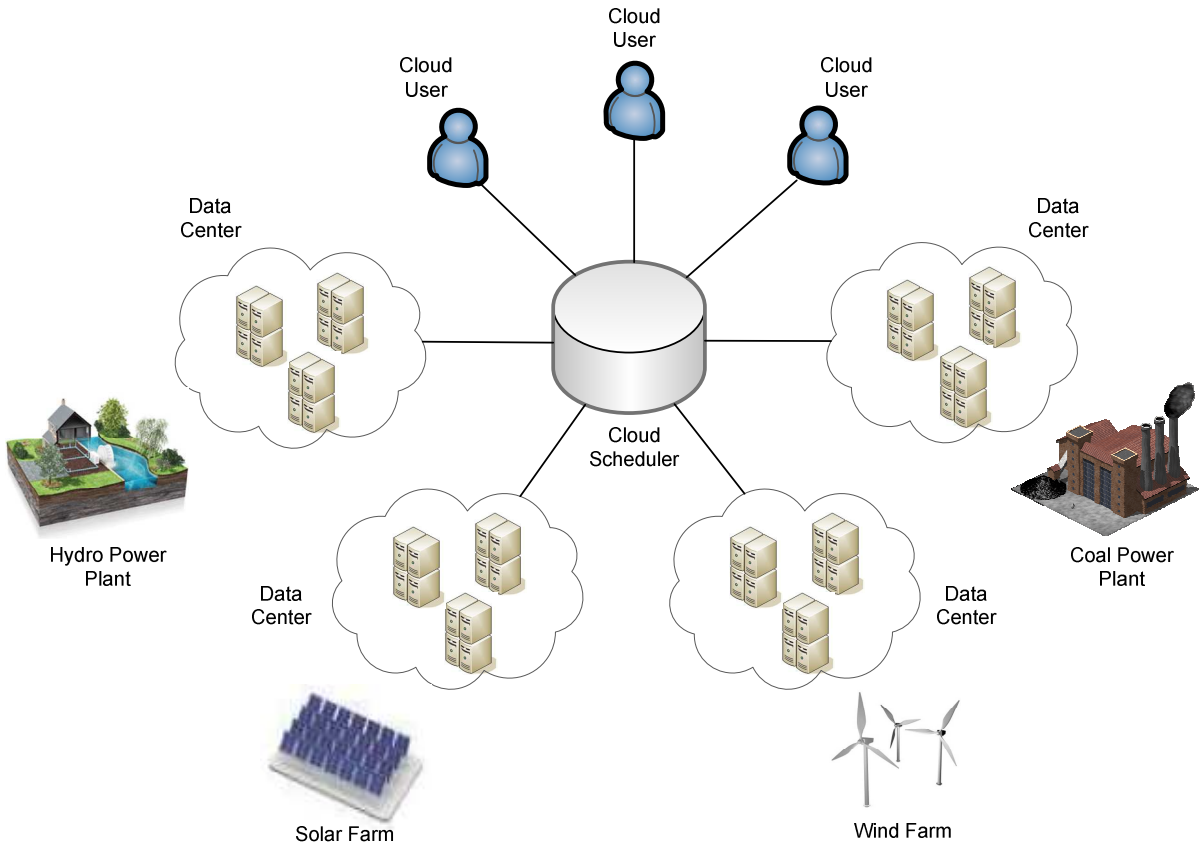


Figure 12: Public Cloud with Central Scheduler

In short, our particular optimization problem is to distribute user workloads to data centers and time slots so that the financial cost (i.e. electricity cost) and environmental cost (i.e. carbon cost) for executing the workloads are optimal. It is worth noting that our framework is agnostic of the chosen cost functions; thus it can incorporate any other optimization objectives.

We continue this chapter by providing a formal specification for our optimization model. We then discuss how to reconcile competing objectives (i.e. electricity cost and carbon footprint) and how to solve the optimization problem. We conclude the chapter by discussing some important factors that dictate the effectiveness of our framework.

4.2 Formulating Optimization Model

Figure 11 depicts our optimization problem. The Cloud scheduler needs to distribute M jobs (workloads) to N data centers and K time slots. Jobs can be distributed in both space, i.e. at which data centers they are executed, and time, i.e. when they are executed. The optimization is subject to multiple constraints (e.g. spare capacity and deadline) and must reconcile multiple competing objectives (e.g., electricity cost and carbon cost).

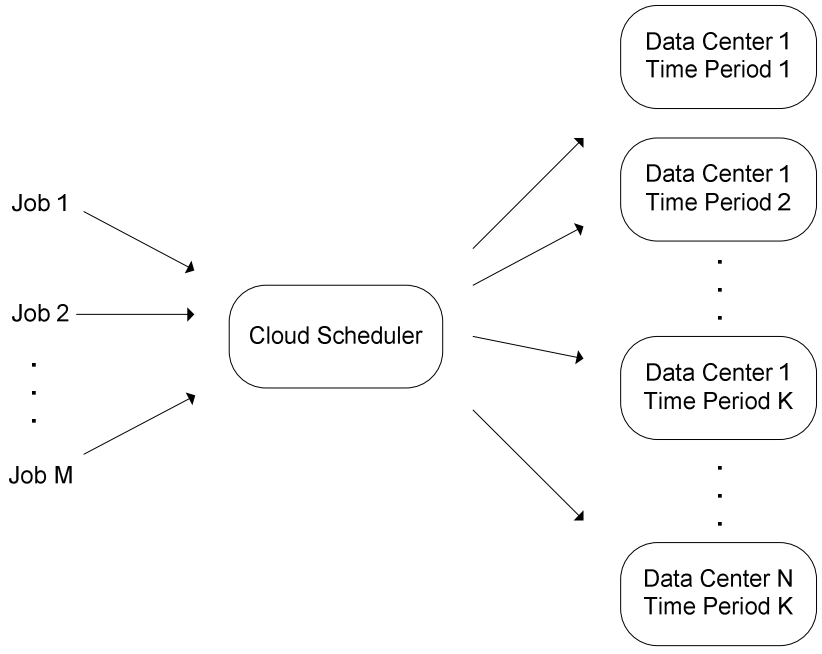


Figure 13: Workload Distribution as an Optimization Problem

The optimization is performed periodically rather than as workloads arrive. At the beginning of each time slot, the Cloud scheduler performs the following functions.

On one hand, the Cloud scheduler receives status updates from individual data centers, detailing how much spare capacity (e.g., the number of available CPUs) each data center has and the associated electricity price and emission intensity for each of the K time slots starting from the current period.

On the other hand, the Cloud scheduler receives workload requirements from Cloud users. The workload requirements detail how many CPU hours a workload takes and its completion deadline. If the users specify their wishes for optimization objectives (e.g., optimizing for financial cost, environmental cost, or any combination of the two), the scheduler will honor user wishes and distribute their workloads accordingly. Otherwise, it will use the default workload distribution strategy defined by the Cloud provider.

Time. We model our workload distribution optimization as a discrete time process. At the beginning of each period (e.g., every minute, every hour), the optimization is performed; mappings are calculated and jobs are distributed to data centers. For jobs that are scheduled for execution in a later time slot, they are still dispatched to data centers but will not run until the chosen time slot.

Workload Types. We focus on CPU-bound computation jobs that require minimal data transfer or storage. The energy profile of a server depends partly on the characteristics of the workloads it runs. Unlike previous work such as (Qureshi et al., 2009) and (Kien Le et al., 2010) which study computationally trivial web hits, we consider workloads that require various amounts of CPU power.

Distribution and Migration. We focus mainly on the initial distribution of user workloads, i.e. when the jobs are first submitted to cloud providers and allocated to some data centers. Data center selection is based on external factors (e.g., electricity price and emission intensity) and internal factors (e.g., data center energy consumption). These factors are expected to change over time.

Electricity prices can go up or down depending on electricity supply-demand imbalance. Emission intensities can fluctuate due to the availability of renewable sources in local regions. The energy consumption of a data center can vary due to changes in existing load or system configuration. To sustain savings over time, long-lived workloads may need to be migrated from one data center to another in response to new conditions. Thus, our work provides a starting point for research on energy-aware workload migration across data centers such as (Buchbinder et al., 2011).

Optimization Formulation

There are M jobs that need distributing. The requirements for job i is defined in the form of a tuple (t_i, $r_{i1},\ldots,r_{iN}$). $\mathbf{t_i}$ is the job deadline, which is defined as the maximum number of time slots allowed before job i must be completed (excluding the waiting time for the optimization process to start). The earliest deadline is one time slot and the latest deadline is K periods. K is the number of time slots with available price & emission signals (e.g., K=2 means price and emission intensity signals are available for the immediate period and next period). $\mathbf{r_{ij}}$ is the number of CPU hours required for job i at data center j.

There are N data centers where jobs can be allocated. The status update that data center j sends to the Cloud scheduler is defined in the form of a tuple ($\lambda_{j1},\ldots,\lambda_{jK}$, $p_{j1},\ldots,p_{jK}$, $q_{j1},\ldots,q_{jK}$). $\boldsymbol{\lambda_{jk}}$ is the number of CPU hours data center j offers at time slot k. $\mathbf{p_{jk}}$ is the electricity price that data center j pays at time slot k; p_{jk} is measured in \$/kWh. $\mathbf{q_{jk}}$ is the emission intensity of the generation mix that supplies electricity to data center j at time slot k; q_{jk} is measured in tCO_2e/kWh.

We seek a data center j in a time slot k to allocate job i so that the electricity cost and carbon cost of executing the job are optimal. Let $\mathbf{g_{ij}}$ be the amount of energy required for running job i at data center j; g_{ij} is measured in kWh. The electricity cost and carbon cost of running job i at data center j in time slot k are defined as follows:

$$\text{Electricity Cost } E_{ijk} = g_{ij} \times p_{jk} \qquad (1)$$

$$\text{Carbon Cost } C_{ijk} = g_{ij} \times q_{jk} \qquad (2)$$

Let $\mathbf{s_{ijk}}$ represent the mapping between job, data center, and time slot; s_{ijk} = 1 if job i is allocated to data center j in time slot k, s_{ijk} = 0 otherwise. Let $\mathbf{h_{ij}}$ be the number of time slots elapsed before job i finishes at data center j. Our optimization model can be defined as follows:

$$\text{SELECT: } \{s_{ijk}: 1 \le i \le M, 1 \le j \le N, 1 \le k \le K\}$$

$$\text{TO MINIMIZE: Electricity Cost } = \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{k=1}^{K} E_{ijk} \times s_{ijk} = \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{k=1}^{K} g_{ij} \times p_{jk} \times s_{ijk}$$

$$\text{Carbon Cost} = \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{k=1}^{K} C_{ijk} \times s_{ijk} = \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{k=1}^{K} g_{ij} \times q_{jk} \times s_{ijk}$$

SUBJECT TO:

i) $\sum_{i=1}^{M} r_{ij} \times s_{ijk} \leq \lambda_{jk} \; \forall j,k$

ii) $h_{ij} \leq t_i \leq K \forall i$

Figure 14: Workload Distribution Optimization Model

The first constraint ensures the total workload allocated to a data center in a given time slot does not exceed its spare capacity. The second constraint ensures that the job deadline is not violated.

Postponement and Deadline. We assume that each job can be completed within one time slot regardless of the number of CPU hours it requires. Thus, the execution of a job can be postponed to a later period, but when the execution starts, it will be completed by end of that period. In a Cloud Computing environment, this can be achieved by scaling up computing resources. A 24-CPU-hour job (r_{ij}=24), for example, can be completed with 1 CPU in 24 hours or 24 CPUs in 1 hour. This assumption simplifies our calculation for the postponement of a job's execution given its deadline.

Job Requirements. We assume that Cloud users are able to specify their workload requirements (t_i, $r_{i1},\ldots,r_{iN}$). This is similar to the way Cloud users acquire virtual machines from Amazon EC2. Users specify in advance how much computing resources (i.e. CPU, storage, bandwidth) they need and for how long. Resource usage is charged on an hourly basis; even when a job finishes in less than one hour, users still need pay for the full hour.

Energy Usage. We assume that all the energy efficiency parameters of a data center such as CPU power-frequency relationship and PUE are known in advance by the Cloud scheduler so that the scheduler can estimate how much energy is needed for executing a workload at a data center. Section 4.1 provides more details about how our energy model is constructed.

User Wishes for Optimization Objectives. For a public Cloud infrastructure operating in multiple regions with various constraints on carbon, its users may have unique wishes for optimization objectives. Some may prefer their workloads to be distributed in such a way that minimizes their financial costs. Other

may prefer a workload distribution that balances between financial and environmental costs so that their accumulated footprints do not exceed their regulated carbon caps, or so as to reduce their carbon taxes or credit purchases on carbon exchanges. Our optimization formula can be applied to individual user workloads by setting M (the number of jobs) to 1. Since the cheapest electricity is often not the least carbon-intensive, certain trade-off would need to be made. We discuss in detail our method for reconciling competing optimization objectives in the next section.

4.3 Reconciling Optimization Objectives

Optimization problems with multi-dimensional cost functions are generally more complicated than problems with single-dimensional cost functions. One particular complication is that multi-objective optimization problems do not always have dominant solutions—it is not guaranteed that there will be a choice of solution that simultaneously improves all the cost components or criteria[1].

The result of the optimization is often a Pareto surface: a set of non-dominant solutions where none of the cost components or criteria can be improved without sacrificing any others. In other words, moving from one Pareto solution to another Pareto solution reduces cost along some dimension(s) but increases it along some other dimension(s). For example, we may be able to reduce the electricity cost by skewing workload distribution towards data centers powered by coal-fired power plants but this will increase the associated carbon footprint.

There are generally two approaches to reconciling competing objectives: sequential optimization and aggregate cost function (Qureshi, 2010). In the following sub-sections, we first describe the two general approaches and discuss some of their advantages and limitations. We then present a novel aggregation method that allows us to capture various interrelationships between criteria.

4.3.1 Sequential Optimization

One common approach is to prioritize the objectives and optimize for one objective after another, starting with the one that has the highest priority. For instance, we could optimize workload distribution for electricity cost first. The obtained optimal electricity cost would then be treated as a constraint when optimizing for carbon footprint. We can relax this process by, for example, weakening the electricity cost constraint (e.g., electricity cost should be no more than 10% than optimal).

[1] We use the two terms cost components and criteria interchangeably. In our context, a criterion is basically a scale of measurement. It is used to measure the relative level of achievement of an objective. In that sense, cost components and criteria refer to the same thing.

An advantage of this approach is that it allows new optimization modules to be added on without affecting existing modules. Introducing a new module for minimizing electricity cost or carbon footprint can be simply a matter of appending it to the end of the optimization pipeline.

A limitation of this approach is that there is no guarantee that all the objectives are taken into consideration. Optimizing for higher-priority objectives might eliminate all but one alternative when the optimization process reaches lower-priority objectives, hence rendering them meaningless.

For Cloud providers who want compromises between electricity cost & carbon footprint, sequential optimization may not be an ideal approach. Given that only one objective can be optimized for at a time regardless of the other objectives, there is little room for compromises.

4.3.2 Cost Aggregation

Another common approach is to aggregate the different cost components to one figure of merit by using an aggregation operator such as arithmetic mean, median, weighted minimum/maximum, or ordered weighted averaging operator (Grabisch, 1996). For instance, we could convert both electricity cost and carbon footprint into the same monetary unit using constant conversion coefficients (i.e. electricity price and carbon price), then use the arithmetic mean to select one from the pool of workload distribution alternatives.

An advantage of this approach is that it effectively maps a multi-dimensional cost function down to a single-dimensional cost function, enabling existing single-objective optimization solvers to be re-used. Another advantage is that it provides a convenient way for decision makers to impose their preferences on optimization objectives. Different compromises can be made by introducing different weights in the aggregate cost function.

A limitation of this approach is that it requires cost components to be normalized as they are often of different magnitudes. For example, electricity cost is measured in dollars and carbon footprint is measured in tons of CO_2e. It does not make sense to place these two costs in the aggregate function without converting them to a common unit of measurement, or more generally, mapping their values into a common scale.

4.3.3 Weighted Sum Aggregation

We propose a weighted sum function for aggregating electricity cost and carbon cost to one figure of merit expressed in dollar terms. The electricity cost is calculated from the wholesale price of electricity.

The carbon cost measured in tCO_2e can be converted into dollar terms using a carbon price. User wishes are expressed as weightings for optimization objectives. By choosing different weightings, decision makers can alter the aggregate cost, hence the distribution of workloads, and achieve different trade-offs.

Let $\mathbf{w_1}$ and $\mathbf{w_2}$ be the weightings for the electricity cost and carbon cost respectively, $\mathbf{u_{jk}}$ be the carbon price that data center j pays in time slot k. The aggregate cost is calculated as follows:

$$\text{Aggregate Cost } A_{ijk} = w_1 \times E_{ijk} + w_2 \times C_{ijk} \times u_{jk} = w_1 \times g_{ij} \times p_{jk} + w_2 \times g_{ij} \times q_{jk} \times u_{jk} \quad (3)$$

$$0 \le w_1, w_2 \le 1$$
$$w_1 + w_2 = 1$$

Thus, our optimization model in Figure 13 can be redefined as follows:

SELECT: $\{s_{ijk} : 1 \le i \le M, 1 \le j \le N, 1 \le k \le K\}$

TO MINIMIZE:

$$\text{Total Aggregate Cost} = \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{k=1}^{K} A_{ijk} \times s_{ijk}$$

$$= \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{k=1}^{K} \left(w_1 \times E_{ijk} + w_2 \times C_{ijk} \times u_{jk}\right) \times s_{ijk}$$

$$= \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{k=1}^{K} \left(w_1 \times g_{ij} \times p_{jk} + w_2 \times g_{ij} \times q_{jk} \times u_{jk}\right) \times s_{ijk}$$

$$= \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{k=1}^{K} \left(w_1 \times p_{jk} + w_2 \times q_{jk} \times u_{jk}\right) \times g_{ij} \times s_{ijk}$$

$$0 \le w_1, w_2 \le 1$$

$$w_1 + w_2 = 1$$

SUBJECT TO:

i) $\sum_{i=1}^{M} r_{ij} \times s_{ijk} \le \lambda_{jk} \; \forall j, k$

ii) $h_{ij} \leq t_i \leq K \forall i$

Figure 15: Workload Distribution Optimization Model with Aggregate Cost

Table 2: Optimization Model Parameters

Symbol	Meaning
s_{ijk}	Equals 1 if job i is allocated to data center j in time slot k. Equals 0 otherwise.
w_1	Weight for electricity cost in the aggregate cost function. Real number between 0 & 1.
w_2	Weight for carbon cost in the aggregate cost function. Real number between 0 & 1.
M	Number of jobs/workloads.
N	Number of data centers.
K	Number of time slots with available price & emission signals (e.g., K=2 means signals for the immediate period & next period are available).
A_{ijk}	Aggregate cost (in $) for running job i at data center j & time slot k.
E_{ijk}	Electricity cost (in $) for running job i at data center j & time slot k.
C_{ijk}	Carbon cost (in gCO_2e) for running job i at data center j & time slot k.
g_{ij}	Energy usage (in kWh) for running job i at data center j.
p_{jk}	Electricity price (in $/kWh) that data center j pays at time slot k.
q_{jk}	Emission intensity (in gCO_2e/kWh) that data center j has at time slot k.
u_{jk}	Carbon price (in $/$gCO_2e$) that data center j pays time slot k.
r_{ij}	Number of CPU hours required for job i at data center j.
λ_{ij}	Number of CPU hours data center j offers at time slot k.

h_{ij}	Number of time slots elapsed before job i finishes at data center j.
t_i	Maximum number of time slots allowed before job i must be completed (i.e. job deadline)

4.4 Performing Workload Distribution Optimization

Having defined the aggregate cost function and constraints, our next step is to find an efficient optimization mechanism. The mechanism would help us obtain optimal solutions, i.e. workload distributions that have optimal costs while not violating any of the constraints. Also, these solutions should be obtained in a timely manner, i.e. the amount of time spent on finding the solutions should be much shorter than the optimization time slot.

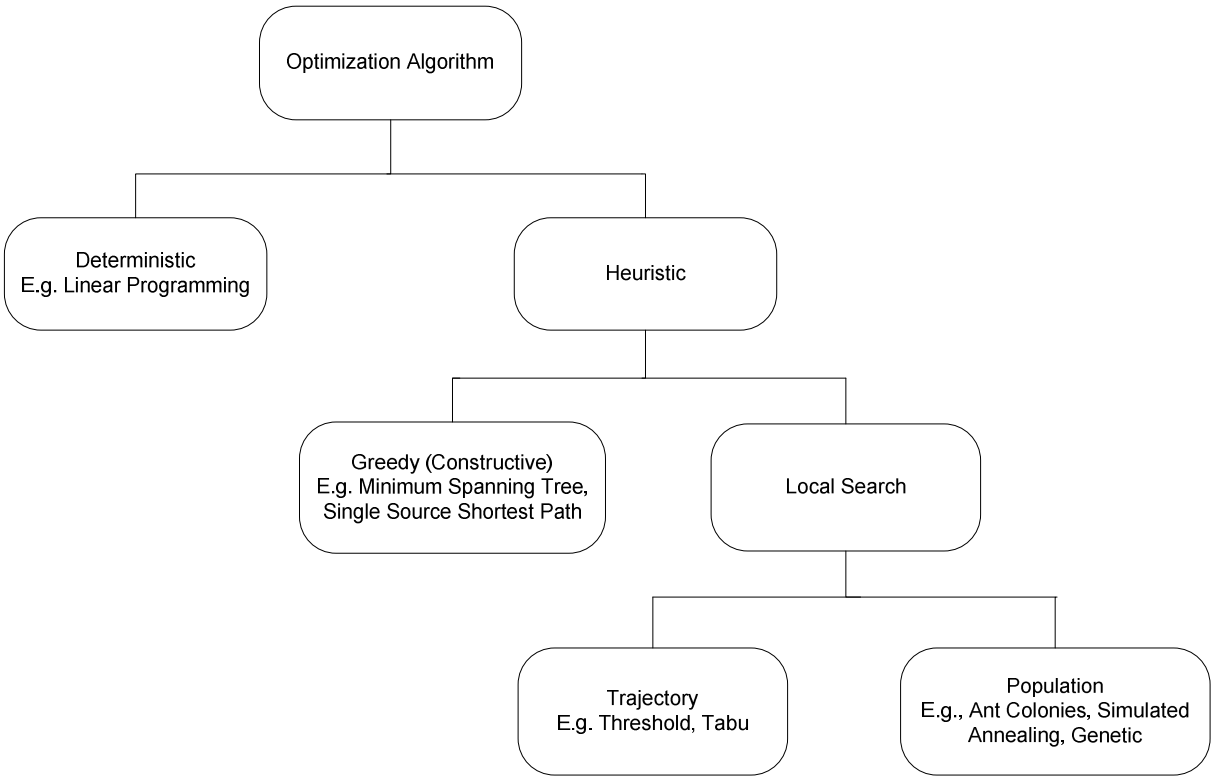


Figure 16: Optimization Algorithm Classification derived from (Gilli & Winker, 2009)

Optimization is a research domain that has been studied rigorously and for which a large number of solutions already exist. Review of the literature on the topic can be found in (Andersson, 2000), (Marler & Arora, 2004), and (Gilli & Winker, 2009).

Algorithms for solving an optimization problem generally fall into one of the two classes: deterministic algorithms and heuristic (or randomized) algorithms.

4.4.1 Deterministic Algorithms

A deterministic algorithm describes a finite calculation procedure to reach the solution of a given problem. Given a particular input, it produces the same output and it does that by executing the same sequence of calculation.

Deterministic algorithms guarantee to produce optimal solutions but they can only apply to problems that satisfy their requirements and constraints. As a result, the original problems often have to be simplified to fit the algorithms. For example, linear programming is only applicable when the objective function to be optimized is linear and all relations among the variables corresponding to resources are linear (Qureshi, 2010).

4.4.2 Heuristic Algorithms

A heuristic algorithm mimics some behavior found in nature to reach the solution, e.g., the principle of evolution through selection and mutation (genetic algorithms), the annealing process of melted iron (simulated annealing) or the self organization of ant colonies (ant colony optimization). Due to their stochastic nature, given a particular input, heuristics may follow different calculation paths and end up with different solutions.

Heuristic algorithms do not guarantee optimal solutions; they finish without knowing whether the outcome is optimal. Their strength, however, lies in their ability to produce high-quality approximations to optimal solutions without relying on any assumptions about the optimization problems; it is sufficient to be able to evaluate the objective function for a given element of the search space (Gilli & Winker, 2009).

Optimization heuristics can be divided into two classes: Greedy (or Constructive) algorithms and Local Search algorithms.

Greedy Heuristics. Greedy heuristics start with an “empty solution” and repeatedly extends the current solution until a complete solution is constructed. An example is Dijkstra's graph search algorithm for solving the Single Source Shortest Path problem. For a given source vertex (node) in the graph, the algorithm finds the path with lowest cost (i.e. the shortest path) between that vertex and every other vertex.

To solve our workload distribution problem using Dijkstra's algorithm, consider each data center in a time slot as a node in an empty graph (i.e. N x K isolated nodes for N data centers and K time slots). For each workload, starting with the heaviest, the heuristic makes it the initial node and calculates the distance between this node and each of the other N x K nodes, excluding those that do not satisfy workload requirements. The distance is the workload execution cost. The path with the lowest cost (i.e. shortest path) represents the optimal mapping between the workload, data center, and time slot. The procedure repeats until all the workloads are mapped (i.e. a complete solution is obtained).

Local Search Heuristics. Local Search heuristics start with a complete solution and tries to improve it by making local adjustments. Local search algorithms can be further divided into two sub-classes: trajectory-based algorithms and population-based algorithms. The former works with a single solution. An example is the Tabu Search heuristic in which visited solutions are kept track of; once a potential solution has been determined, it is marked as "tabu" (i.e. "taboo") so that the heuristic does not visit that possibility repeatedly. The latter updates the whole set of solutions simultaneously. Examples are Genetic, Ant Colonies, and Simulated Annealing.

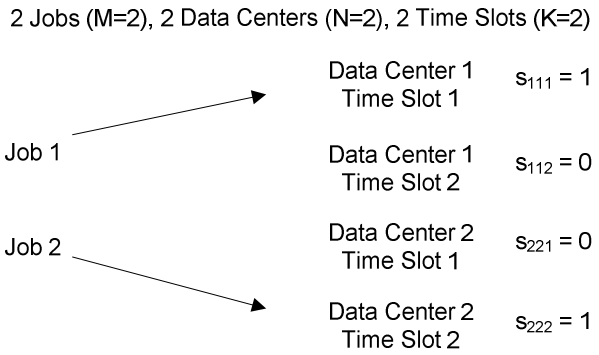


Figure 17: Workload Distribution Candidate

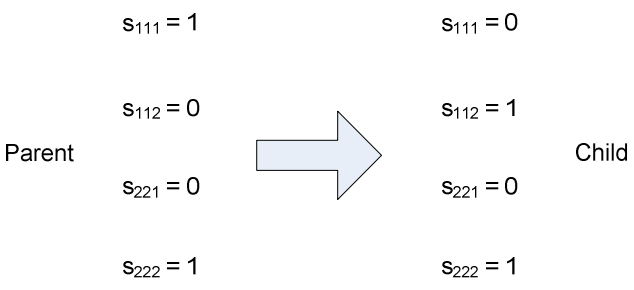


Figure 18: Mutation Operation

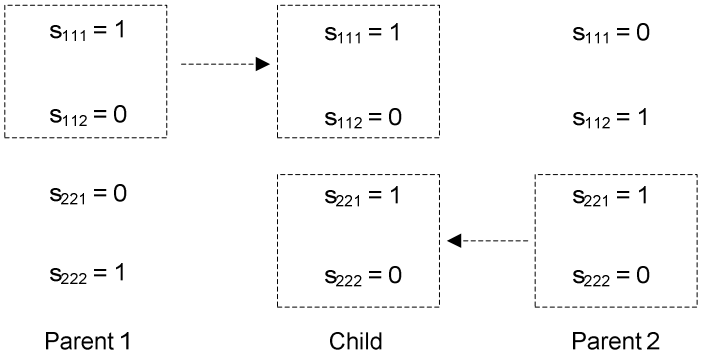


Figure 19: Cross-Over Operation

To solve our workload distribution problem using Genetic algorithm, consider each possible distribution of workloads as an individual in a population. The heuristic starts with an initial population (i.e. a set of workload distribution candidates). It (i) evaluates each individual in the population using the aggregate cost function as the fitness function, (ii) selects the fittest individuals to be the parents for the next generation, (iii) randomly alters and/or combines the parents (i.e. performing evolutionary operators such as cross-over and mutation) to produce children, and (iv) inserts the children back into the population. The procedure then starts over again. The heuristic finishes after a certain number of iterations or when certain conditions are met. The workload distribution that has the lowest total aggregate cost is considered as the best distribution.

4.4.3 Workload Distribution Optimization Heuristics

As described in the problem specification, we optimize workload distribution for multiple objectives and under multiple constraints. Instead of designing highly-specialized deterministic algorithms, we propose the use of heuristic algorithms for their ability to produce high-quality solutions for various problem instances without having to enumerate every possible candidate in the solution space. Moreover, we choose greedy heuristics over local search heuristics as local search heuristics are often associated with high computational costs and are harder to implement (Andersson, 2000).

We propose six greedy heuristics, all based on Dijkstra's graph search algorithm (see section 4.4.2) but have different optimization objectives. Also, three of them perform look-ahead on electricity price and emission intensity (i.e. evaluating K time slots instead of just the immediate time slot) while the other three do not. By comparing heuristics that have different objectives (i.e. different weights for different

objectives), we demonstrate how our framework can incorporate multiple objectives and how it can handle the trade-offs between those that conflict with each other. By comparing heuristics with and without look-ahead, we evaluate potential savings from exploiting temporal variations in electricity price and emission intensity.

Table 3: Comparison of Workload Distribution Optimization Heuristics

Heuristic	Optimization Objective(s)	With Look-Ahead
Greedy-Electricity-Cost (GEC)	Electricity Cost $(w_1 = 1, w_2 = 0)$	No
Greedy-Carbon-Footprint (GCF)	Carbon Footprint $(w_1 = 0, w_2 = 1)$	No
Greedy-Both (GB)	Electricity Cost & Carbon Footprint $(w_1 = 0.5, w_2 = 0.5)$	No
Greedy-Electricity-Cost-With-Look-Ahead (GECwLA)	Electricity Cost $(w_1 = 1, w_2 = 0)$	Yes
Greedy-Carbon-Footprint-With-Look-Ahead (GCFwLA)	Carbon Footprint $(w_1 = 0, w_2 = 1)$	Yes
Greedy-Both-With-Look-Ahead (GBwLA)	Electricity Cost & Carbon Footprint $(w_1 = 0.5, w_2 = 0.5)$	Yes

4.5 Important Factors

In practice, the effectiveness of our optimization framework depends on many factors. Some of them dictate whether workloads should be distributed across data centers (energy proportionality, spare capacity, etc.). Others influence how dynamic shifting of workload can reduce costs while maintaining

quality of service (discrete time process, etc.). The following list is not meant to be exhaustive but rather a starting point for future discussion.

Energy Proportionality. Energy proportionality is the degree to which the electrical energy consumed by a data center depends on the amount of workload placed on it. For computing clouds whose data centers are not energy-proportional, our optimization framework would be minimally effective. The reason is because these infrastructures always consume an amount of energy that is close to their energy peak demand regardless how much workload they actually have; thus, shifting workload around wouldn't make much difference.

The degree of energy proportionality of a data center depends on how energy-proportional its subsystems are. As discussed in section 2.2.2, servers have the largest share of energy consumption in a data center. An energy-proportional server should only spend as much energy as the given load. While this goal is yet to be achieved, significant progress has been made in the area of system engineering to reduce server idle waste (Qureshi, 2010) (Varsamopoulos, Abbasi, & Gupta, 2010). Thus, energy proportionality computing will soon become a reality.

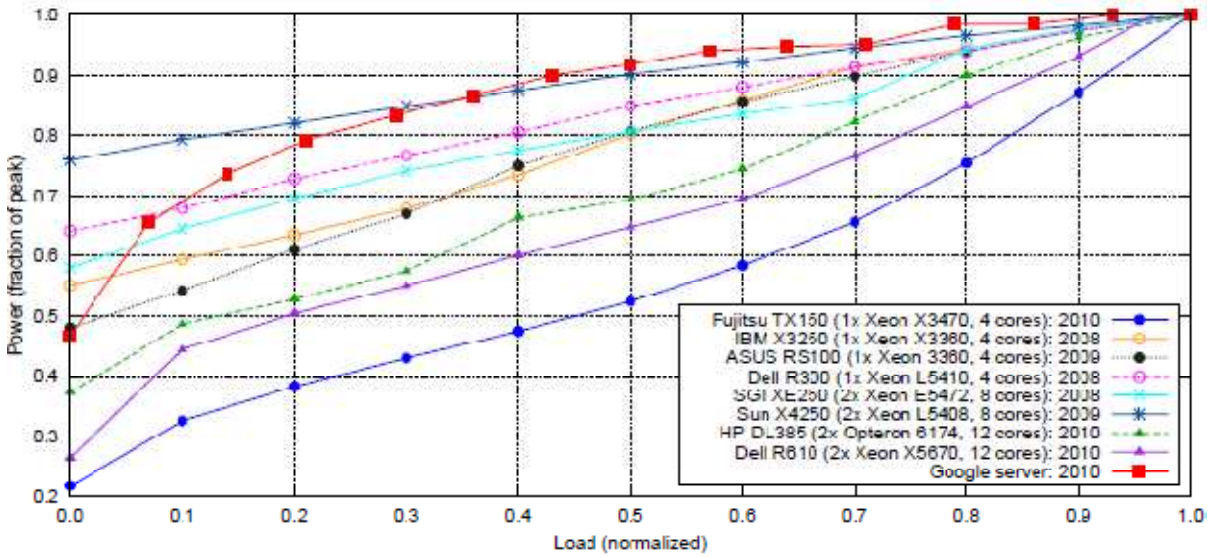


Figure 20: Server Power Consumption vs. Load (Qureshi, 2010)

Spare Capacity. Our optimization heuristics aim to shift workload away from data centers at high-cost locations to data centers at low-cost locations. Such a strategy can only work when there is enough spare capacity to accommodate shifted workload.

Fortunately, real-world systems spend most of their time operating below their maximum capacity. Google, for instance, has reported that their servers mostly run at 10 to 50 percent of their maximum utilization levels (Hoelzle & Barroso, 2009). This is a design by choice as service providers, especially Cloud providers such as Google and Amazon, must provision for peak load to ensure that the quality of their services does not degrade under heavy load at any time.

There are strong incentives for service providers to find useful work for their servers at idle time. When idle times coincide with low electricity prices and/or emission intensities, the incentives are even stronger. While previous work such as (Qureshi et al., 2009) and (Garg et al., 2011) only advocates for shifting workloads in space, we propose shifting workloads in time to capture more savings opportunities.

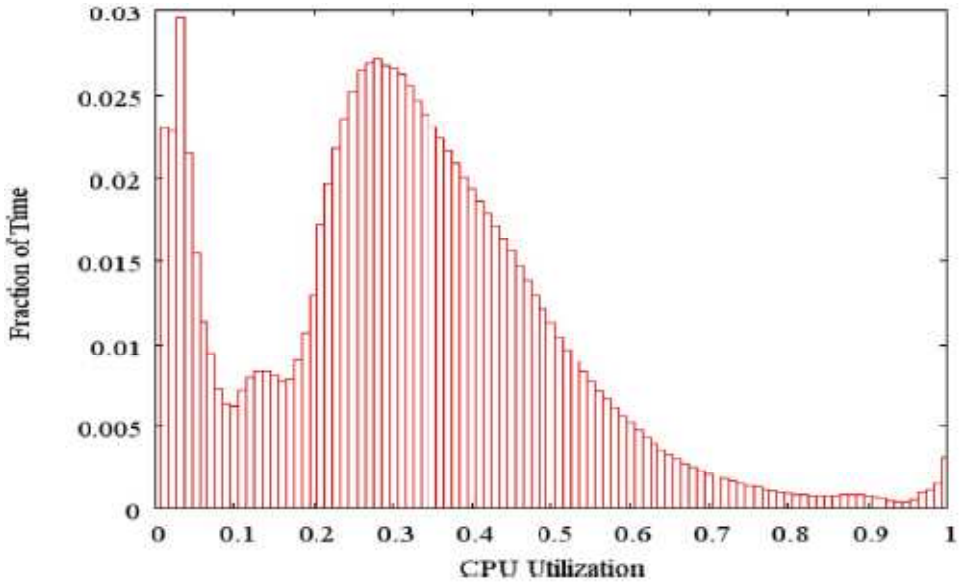


Figure 21: Activity profile of a sample of 5,000 Google servers over a period of 6 months (Hoelzle & Barroso, 2009)

Impact on Local Markets/Grids. Shifting workloads from one geographical region to another means shifting the power demand between those regions. As power demand rises or falls, the balance between

power demand and supply is adjusted, which may trigger changes in electricity price and emission intensity. This feedback loop effect is rather complex and difficult to quantify.

In regions with wholesale electricity markets, electricity prices reflect the current state of power demand and supply in the regions. When workload is shifted from region A to region B, the power demand falls in region A and rises in region B. Given the scale of data center energy use (see section 2.2.1), such workload migration can lead to an electricity price rise in region B and a price fall in region A, affecting not only the involved data centers but also other energy consumers in the two regions.

The emission intensity also changes as power supply changes. In regions whose base load is handled by hydro power plants while peak load is handled by fossil-fueled power plants, emission intensities increase as more peaking power plants are brought online to handle surges in demand. In regions whose base load is handled by coal-fired power plants, however, the addition of renewable energy sources in response to demand surges helps bring down the emission intensity of those regions.

For simplicity, we assume data centers have no effects on the electricity market or generation mix of the regions where workload shifting occurs. This assumption is acceptable because the power usage of individual data centers is still small in comparison to the entire power requirements of their geographical regions.

Price and Emission Signals. Our framework enables Cloud providers to distribute workloads optimally based on the electricity prices and emission intensities of the regions where their data centers are located. An underlying assumption is that the price and emission signals are available to Cloud providers.

Electricity prices in Real-Time (Spot) markets are posted by market operators in real time. Emission intensities are currently not available, but they can be derived from generation records. We provide a formula for calculating emission intensity based on generation output by fuel type in section 4.2.3. Many market operators also provide day-ahead or hour-ahead forecast on price and power demand. IESO (Ontario market operator) posts day-ahead price and demand forecast (IESO, 2011b). AESO (Alberta market operator) posts hour-ahead price forecast (AESO, 2011a). CAISO (California market operator) posts both day- and hour-ahead demand forecast (CAISO, 2011d). Forecast on price can be used as price signal. Forecast on demand can be used to estimate supply and combine with information about generation mix (e.g., fuels used for base load, intermediate load, and peak load) to derive emission intensity. What is more important to the optimization model is the differentials of electricity

price and emission intensity between regions rather than the exact values of electricity price and emission intensity of each region.

Forecasting is a complex topic and beyond the scope of this study. Here we focus on the matter of distributing workloads based on external cost signals, which include current and forecast electricity prices and emission intensities. We assume that these signals are available to the Cloud provider.

Workload Scalability. In our study we assume that any workload can be finished within one time slot by pouring in more CPU power. The assumption allows us to postpone workload execution to the last possible time slot and still be able to guarantee that the job deadline is not violated. For this strategy to work there must be sufficient CPU power available to speed up the execution. Also, the workload must be scalable in the sense that its execution can be split up among multiple CPUs; thus, the more CPUs added, the faster the job gets done.

In practice, it is not always possible to scale a workload horizontally since many existing algorithms are designed for sequential execution. With the wide spread of multi-core CPUs, however, new algorithms are emerging to take advantage of new hardware architectures. Google's MapReduce, for instance, allows users to turn a key/value pair into a set of intermediate key/value pairs which can be processed in parallel and combined into the final output in the end (Dean & Ghemawat, 2008). Thus, it is possible to have scalable workloads.

5 Energy Modeling

The Cloud scheduler must be able to estimate how a data center's energy usage would change when the workload placed on it increases or decreases. How close to optimal the workload distribution is depends partly on the accuracy of the energy model.

We model the energy consumption of a data center as a discrete function of its power draw over an accounting period (e.g., an hour). *Power* in our context refers to the average power over a period of time rather than the instantaneous power.

Building an accurate power consumption model for data centers is a non-trivial task. Data centers are highly complex systems with many interacting mechanical and electrical sub-systems. The amount of power drawn by the cooling system, for instance, depends on various factors, from the temperature of the server room to the algorithms being used for load balancing servers. In addition, the design of the data center is evolving rapidly with monolithic designs giving way to newer modular container-based alternatives (Hoelzle & Barroso, 2009). Our goal here is to derive from the literature a composite model that can predict the amount of power consumed by a data center based on its current load.

Ideally Cloud providers would instrument their data centers with power meters to obtain accurate measurement. Direct power measurement can be expensive, however, when done at a fine-grained level (e.g., at individual servers or server components). Power modeling is an alternative approach where various software and hardware counters are taken as input to a mathematical model that predicts power consumption after sufficient training on an appropriately instrumented platform (McCullough et al., 2011).

We focus on the primary sources of energy consumption in the data center (see section 2.2.2). We divide them into two categories, namely IT and non-IT. IT sources include servers, storage, and networking devices. Non-IT sources include cooling (CRAC, chillers, etc.) and power distribution (UPS, PDU, etc.). We assume that other sources of energy use such as lighting only draw a small and fixed amount of power regardless of data center load conditions and hence can be safely ignored.

In short, we model IT power as a non-linear function of server CPU utilization and non-IT power as a fixed fraction of IT power. In the sections that follow, we outline how these sub-systems consume electricity and model the relationship between each sub-system and the workload placed on the data

center. We then assemble these individual models into a unified model that helps us predict the power draw of the whole data center.

5.1 IT Model

Similar to (Qureshi, 2010) we divide IT equipment into two categories. The first category includes servers and storage systems. The second category includes networking devices.

For simplicity we assume that servers have homogeneous hardware and storage systems are simply servers with specialized roles. Furthermore, we assume that all servers in the data center are interchangeable, i.e. workload can be executed on any of the servers that have sufficient spare capacity.

We model networking devices as having constant and relatively small power consumption (e.g., less than 5% of total IT power). Studies such as (Chabarek et al., 2008) and (Mellah & Sanso, 2009) have shown that the power drawn by network switches is largely independent of their utilization; power consumption in idle mode is almost the same as in active mode.

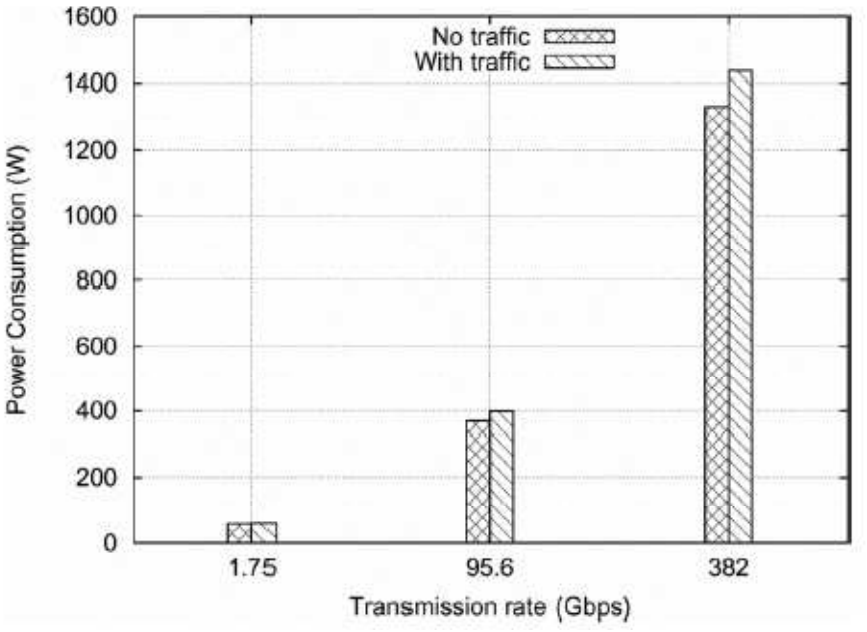


Figure 22: Power Consumption of Routers with and without Traffic (Mellah & Sanso, 2009)

Thus, we model IT equipment as a collection of homogeneous interchangeable servers with some fixed power overhead due to networking devices.

Server Model

A naive model of server power consumption would use the rated power provided by vendors (i.e. name-plate value). A server with a rated power of 750 watts would be assumed to draw as much power once turned on. This assumption is not valid in practice, however. Vasan *et la.* (2010) study the impact of workload on power consumption of 20 different servers and find that even the maximum measured power is significantly lower than the rated power. Thus, the rated power is not a good estimate for the actual power.

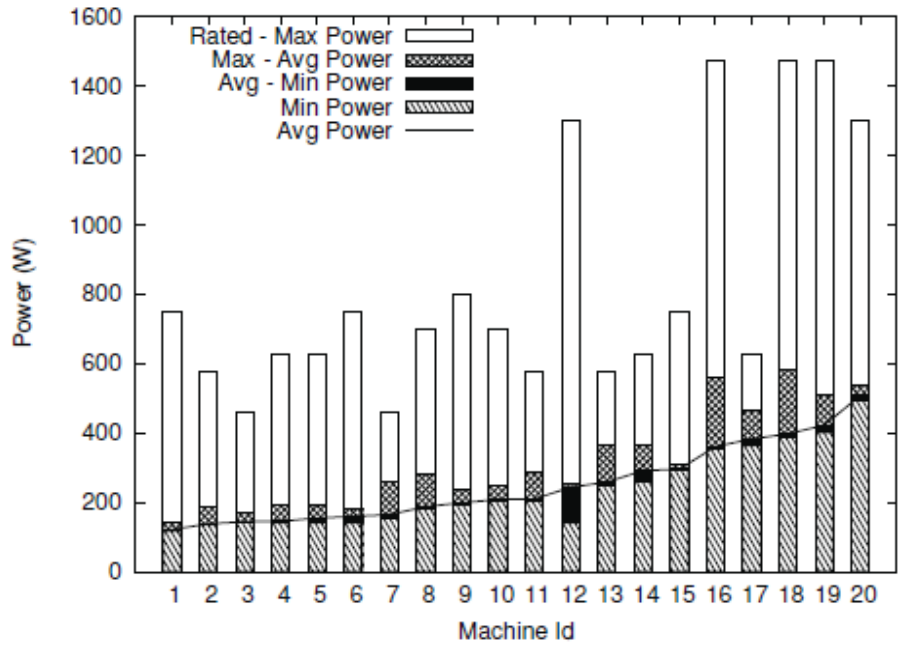


Figure 23: The minimum, maximum, average, and rated power consumptions of servers (Vasan, Sivasubramaniam, Shimpi, Sivabalan, & Subbiah, 2010)

Some regression models have been shown to work well in restricted settings, for example, when the system's static power consumption is dominant and the dynamic component is within the margin of error. McCullough *et la.* (2011) evaluate 5 regression models commonly used in the literature including MANTIS, Lasso regression, Polynomial with Lasso, Polynomial plus exponential with Lasso, and Support vector regression. The first two models are linear, the others are non-linear. The authors find that all the models predict total system power reasonably well. The mean prediction is between 1-3% when the test system runs with a single core and 2-6% when the test system runs with multiple cores. However, these models often perform poorly when predicting sub-system power. The mean relative error is 2-6% for single-core CPU and 10-14% for multi-core CPU with the worst case being 150%. Thus, for servers that

have wide dynamic power ranges and run various CPU-bound workloads, these linear-regression models may not be effective.

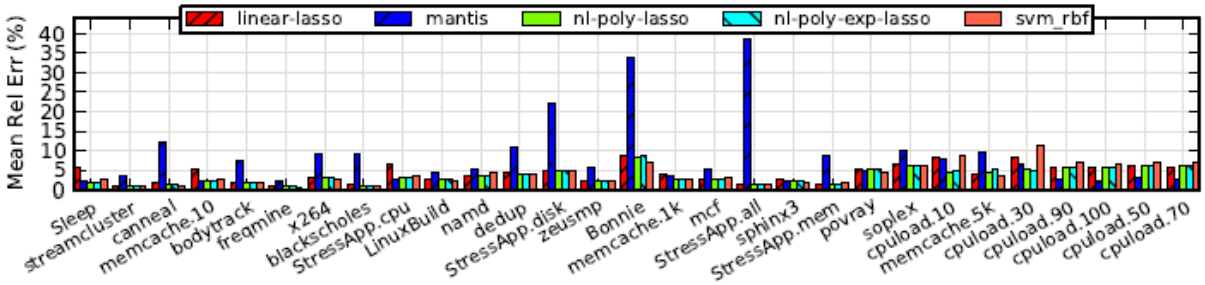


Figure 24: Modeling accuracy for CPU power (single-core), mean relative error is 2-6% across workloads (McCullough et al., 2011)

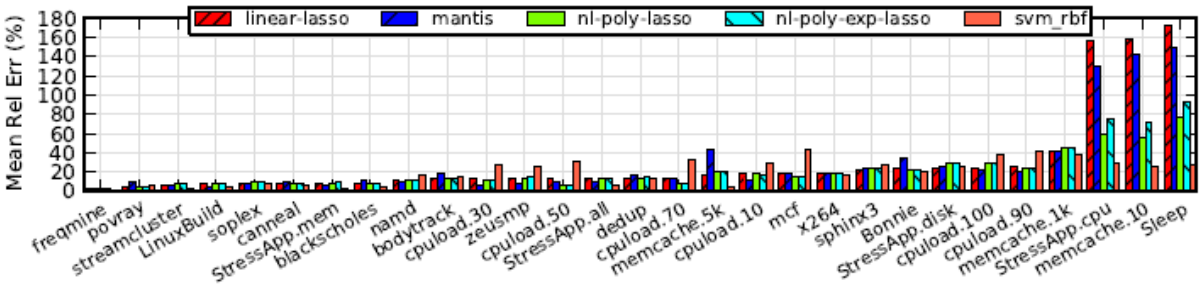


Figure 25: Modeling accuracy for CPU power (multi-core), mean relative error is 10–14% across workloads, but as high as 150% for some workloads on the right (McCullough et al., 2011)

We adopt the model used in (Garg et al., 2011) which computes the energy use of server CPU (single-core). Let $\boldsymbol{\beta}_j$ is the static power measured in watts (W), $\boldsymbol{\alpha}_j$ is the proportionality constant, $\mathbf{f}_j$ is the maximum frequency of CPU at data center j measured in gigahertz (GHz), and $\boldsymbol{\delta}$ is the length of a time slot measured in seconds. For job i that requires r_{ij} CPU hours, the amount of energy consumed by CPU to complete the job in one time slot is calculated as follows:

$$g_{ij}^{IT} = \left(\beta_j + \alpha_j \times f_j^3\right) \times 10^{-3} \times \frac{3600}{\delta} \times r_{ij} \qquad (4)$$

This model is acceptable for two reasons. First, the power draw of a server varies depending on the type of workload it executes. Since we only consider CPU-bound workloads and the CPU draws the most power in a server, it is sufficient in our study to calculate only CPU energy use. Second, our focus is on

exploiting regional differences across data centers rather than how to save energy locally within a data center whereby energy can potentially be saved through other components in a server.

Server Load Distribution Policy

The way load is distributed across servers plays a role in defining the power consumption characteristics of a data center. Skewing load to concentrate it on the smallest possible number of servers and putting unused servers in power-saving modes help reduce idle waste. Spreading load evenly, on the other hand, tends to result in better performance, e.g., smaller latencies and higher reliability (Qureshi, 2010).

We employ an energy-efficient distribution policy in which load is not evenly spread but rather saturates each server, one after another. Servers with no load are turned off or switched to power-saving modes. Energy conservation techniques such as PowerNap (Meisner, Gold, & Wenisch, 2009) can be used to ensure that inactive servers can be brought online quickly to handle new load.

5.2 Non-IT Model

Non-IT equipment includes cooling (CRAC, chillers, etc.) and power distribution systems (UPS, PDU, etc.). Section 2.2.2 provides a brief description of these systems. In short we can model non-IT power as either fixed or a function of IT power, which is more realistic.

Cooling. The cooling system's power consumption varies depending on many factors such as the amount of heat it needs to evacuate, external air temperature, server-room airflow, and the nature of hotspots in the server room (Qureshi, 2010). Modern data centers employ variable speed chillers and fans, enabling their cooling systems to have a wide dynamic range.

Under heavy load, CRAC and chillers can account for more than a third of a data center's peak power draw. Chillers are the dominant energy consumers in a cooling system; they can use three times as much as the other cooling components (Qureshi, 2010). Newer data centers try to reduce the amount of time chillers need to run by exploiting low outside temperatures.

Power Distribution. The efficiency of a power distribution system is related to the electrical load placed on it. The power distribution system imposes a significant overhead. Even in recently built data centers, losses due to distribution can account for more than 10% of the total electricity (Qureshi, 2010).

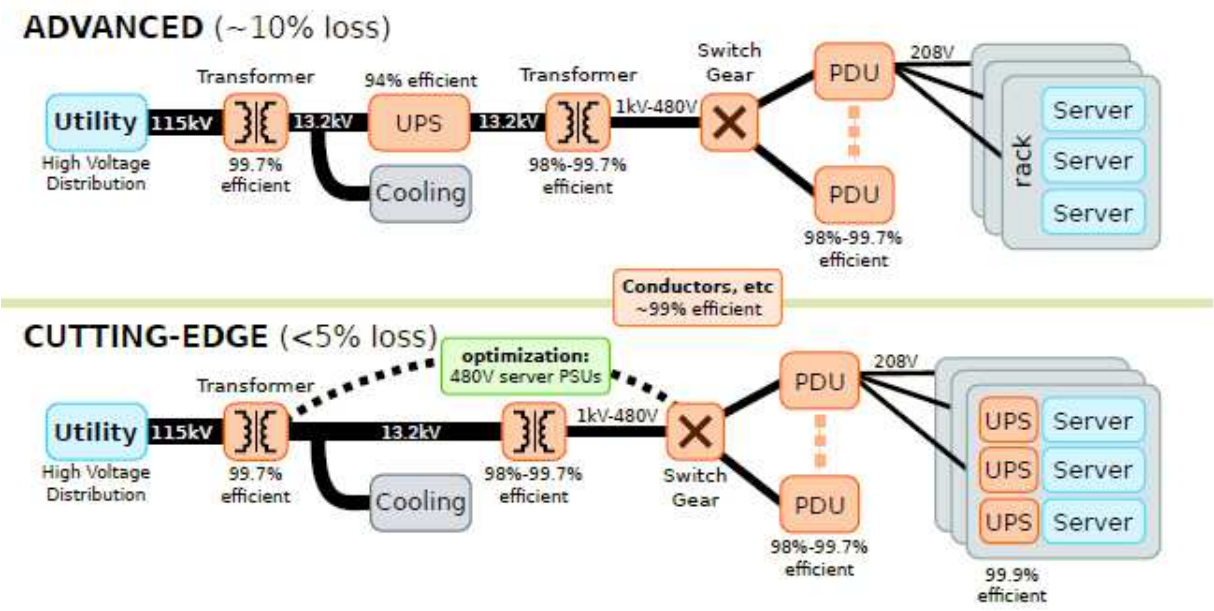


Figure 26: Schematics for two representative power distribution systems; the top design is based on Amazon, the bottom design is based on Google (Qureshi, 2010)

We use the Power Usage Effectiveness (PUE) metric to model non-IT power as a function of IT power. Let μ_j be the PUE of data center j, the amount of energy consumed by non-IT equipment to complete job i is calculated as follows:

$$g_{ij}^{NonIT} = g_{ij}^{IT} \times (\mu_j - 1) \qquad (5)$$

5.3 Data Center Model

The energy usage of a data center is the total amount of energy consumed by its IT and non-IT equipment. For job i that requires r_{ij} CPU hours, the amount of energy consumed by data center j to complete the job in one time slot is calculated as follows:

$$\text{Energy Usage } g_{ij} = g_{ij}^{IT} + g_{ij}^{NonIT} = \left(\beta_j + \alpha_j \times f_j^3\right) \times 10^{-3} \times \frac{3600}{\delta} \times r_{ij} \times \mu_j \qquad (6)$$

The ratio 3600/δ allows us to scale up CPU power if the length of the time slot is less than an hour (3600 seconds), or scale down otherwise. For example, if the time slot is half an hour long (δ=1800), we would need twice as much CPU power to complete the job in one time slot (i.e. 2 x r_{ij}). If the time slot is two hour long (δ=7200), we would need half as much (i.e. 0.5 x r_{ij}).

Table 4: Energy Model Parameters	
Symbol	Meaning
g_{ij}	Energy usage (in kWh) for running job i at data center j.
g_{ij}^{IT}	Energy usage (in kWh) of IT equipment for running job i at data center j.
g_{ij}^{NonIT}	Energy usage (in kWh) of non-IT equipment for running job i at data center j.
β_j	Static power (in W) of CPU at data center j.
α_j	Proportionality factor of CPU at data center j.
f_j	Frequency (in GHz) of CPU at data center j.
δ	Length (in seconds) of a time slot.
μ_j	Power usage effectiveness (PUE) of data center j.
r_{ij}	Number of CPU hours required for job i at data center j.

6 Data Collection

In chapter 4, we develop our workload distribution optimization framework with two main goals in mind. The first goal is for the framework to address the trade-off problem between electricity cost and carbon cost. The second goal is for the framework to achieve cost savings through exploiting temporal and geographical variations in electricity price and emission intensity, i.e. shifting workloads in time and space.

To evaluate our optimization framework with respect to these goals by means of simulation, we need a realistic workload trace, one that provides sufficient detail about computation requirements and arrival patterns of user workloads. We also need empirical data on electricity prices and emission intensities. The data must provide sufficient detail about how prices and emission intensities vary in time and space.

6.1 Workload Trace

To understand the behavior of real compute-intensive workloads, we use a workload trace from the Parallel Workloads Archive (PWA). The workload trace was graciously provided by Moe Jette and maintained by Dror Feitelson (Feitelson, 2008). The complete trace contains 5 months worth of accounting records from a large Linux cluster called Thunder installed at Lawrence Livermore National Lab (LLNL). The Thunder cluster was considered a "capacity" computing resource, meaning that it was intended for running large numbers of smaller to medium computational jobs. Thus, its trace fits the objective of our evaluation.

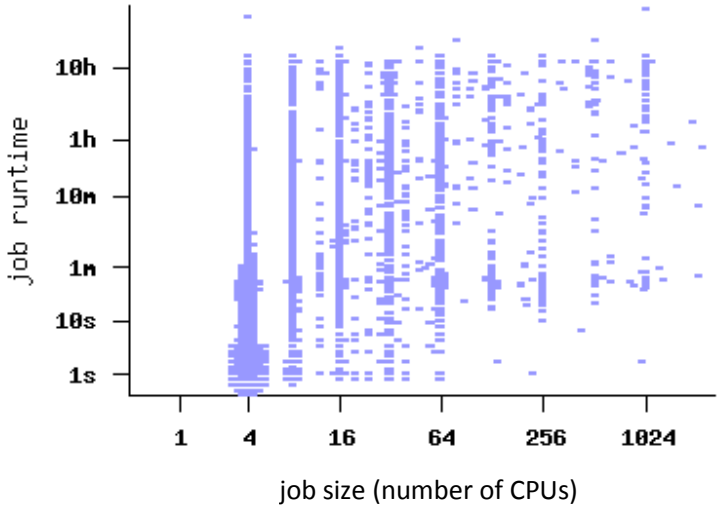


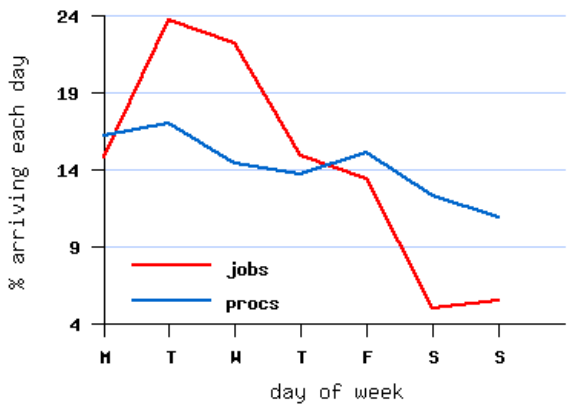
Figure 27: Workload characteristics (Feitelson, 2008)

The Thunder cluster is comprised of a total of 1024 nodes. All nodes are 4-CPU, 1.4 GHz Intel Itanium 2 nodes, which are rack-mounted in frames and managed from a single control machine. Compute nodes are configured into one of two partitions: one for small, short jobs and the other for parallel batch production runs. Compute nodes are not shared with other users; when a job runs, the allocated nodes are dedicated to the job submitter.

Figure 28 shows how job arrival varies in time. Daily and weekly periodicities can be recognized; there are more jobs arriving during the day and week days than the night and weekends. Monthly pattern is less obvious.

The acquired log file is in the standard workload format (swf). Each job is represented by a single line in the file. Lines contain a predefined number of fields, which are mostly integers, separated by whitespace. There are 18 data fields in total. Fields that are irrelevant for a specific log or model are set to -1. Out of these 18 fields, we extract the submit time, requested number of CPUs, and actual runtime of jobs. From the requested number of CPUs and job runtime we derive the number of CPU hours required for each job. Since the log does not contain job deadlines, we assign a random deadline to each job. Each deadline is an integer in the range 1 to 24. A 1-hour deadline means the job must be completed within the immediate hour. A 24-hour deadline means the job must be completed within 24 hours including the immediate hour.

We extract 81-day worth of records from the complete trace. The time period is from Feb 1 to April 22, for which we have data on electricity price and emission intensity (see section 6.2 & 6.3).



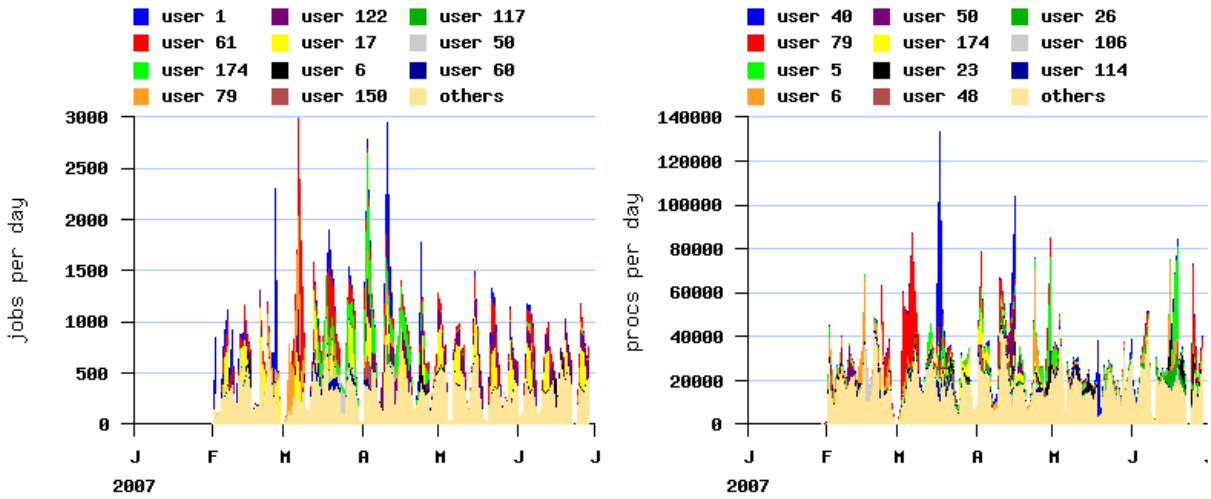


Figure 28: Job arrival patterns (Feitelson, 2008)

6.2 Electricity Price

In regions with electricity monopolies, these firms own all the primary components of electricity supply—generation, transmission, distribution, and retail supply—and have exclusive franchises to supply electricity to end-users at regulated prices. In regions with competitive wholesale markets, generators compete with each other to offer their electricity output to retailers. Retailers then re-price the electricity and sell it to end-users.

The bulk power system in North America is divided into eight reliability regions with various degrees of inter-connectivity (NERC, 2011). Each regional grid is managed by a pseudo-governmental body, a Regional Transmission Organization (RTO). RTOs provide central authority that manages the flow of electricity between generators and consumers and ensure the short-term reliability of the grid.

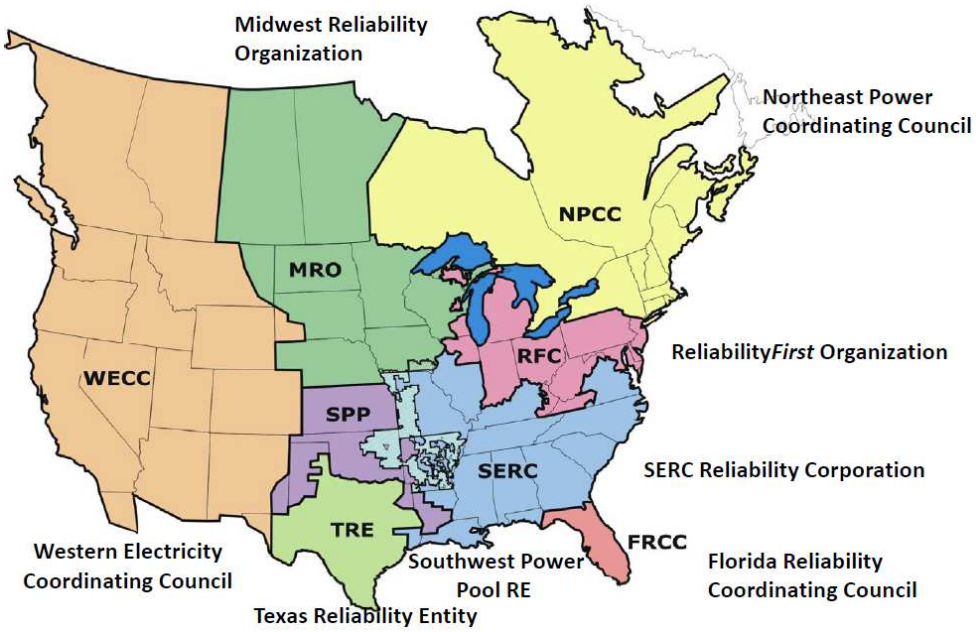


Figure 29: North American Reliability Regions (NERC, 2011)

RTOs also administer wholesale electricity markets where market participants can trade contracts for the delivery of electricity at some specified hour. Prices for these contracts are often determined through an auctioning mechanism: generators present supply offers, consumers present demand bids, and the administering body sets prices and coordinates the delivery of electricity. While bilateral contracts still account for much of the electricity delivered over the grid, wholesale electricity trading has been growing rapidly, and currently covers about 40% of total electricity usage (Qureshi, 2010).

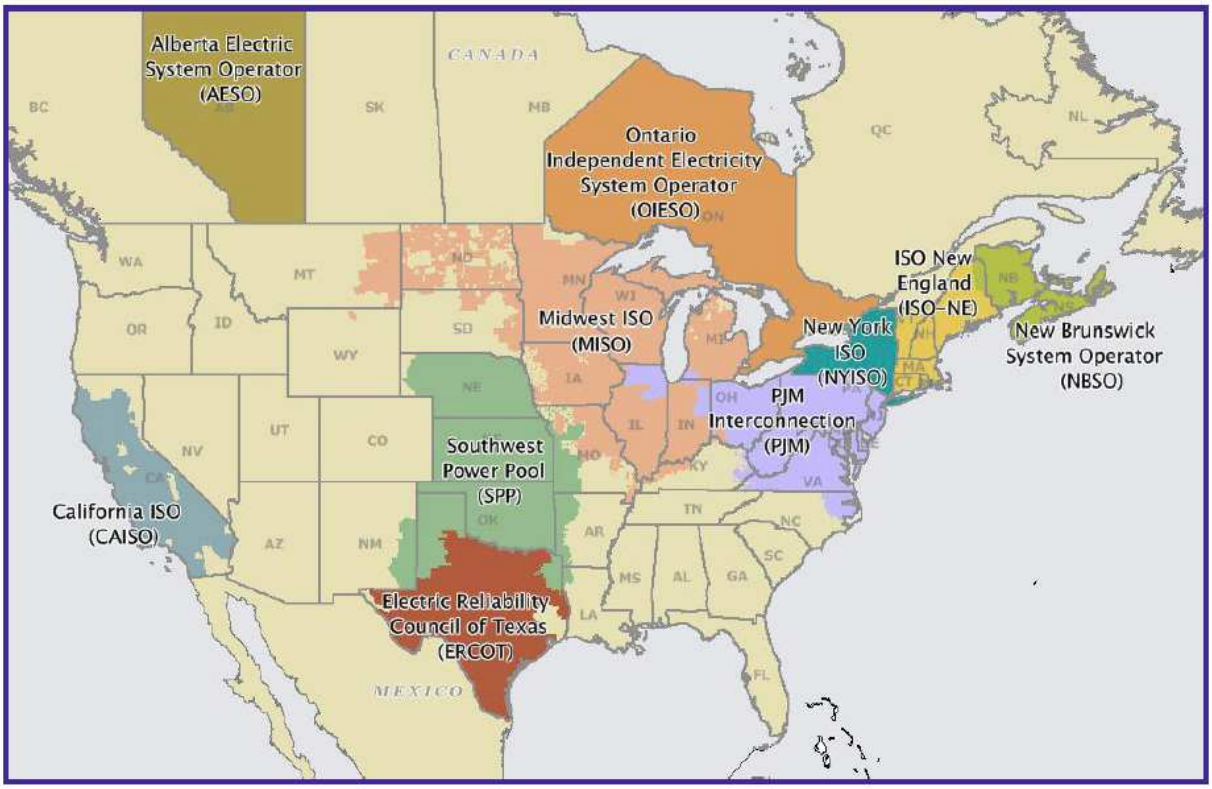


Figure 30: North American Electricity Market Oversight (FERC, 2011)

Locational Marginal Pricing (LMP) is commonly used in North American wholesale markets. The basic idea behind LMP is to determine for a given set of system conditions (generator bids, load profile for the study period, network topology and its limits), the minimum cost of generation that will be required to supply an incremental load of 1 MW at a given location in the network without the violation of any transmission constraints (Hamoud & Bradley, 2004). The locational marginal price at a specific location is the sum of the cost of generating the next MW to supply load at a specific location (based on marginal generation cost), the cost of transmission congestion, and the cost of losses. Thus, LMP provides price signals that account for the additional costs of electricity caused by transmission congestion and line loss at various points on the electricity grid.

Day-ahead market and real-time market are two common types of wholesale electricity markets. In day-ahead markets (forward markets), hourly LMPs are calculated for the next operating day based on anticipated demand, supply, and system conditions. In real-time markets (spot markets), LMPs are calculated at 5-minute or 15-minute intervals based on actual demand, supply, and system conditions. Table 5 provides a list of the competitive wholesale electricity markets in North America.

Table 5: Wholesale Electricity Markets in North America			
Market	Reliability Region	Market Type	Average Spot Price[2]
Ontario (IESO)	NPCC	Real-Time	CA$0.05167/kWh
Alberta (AESO)	WECC	Real-Time	CA$0.09239/kWh
California (CAISO)	WECC	Real-Time, Day-Ahead	$0.05227/kWh
Midwest (MISO)	MRO	Real-Time, Day-Ahead	$0.05117/kWh
New England (ISO-NE)	NPCC	Real-Time, Day-Ahead	$0.08391/kWh
New York (NYISO)	NPCC	Real-Time, Hour-Ahead, Day-Ahead	$0.09009/kWh
PJM	RFC and SERC	Real-Time, Day-Ahead	$0.07175/kWh
SPP	WECC	Real-Time	$0.05742/kWh
Texas (ERCOT)	TRE	Real-Time, Day-Ahead	$0.07052/kWh

We collected 81 days worth of hourly Real-Time market prices from public data archives maintained by market operators. The dataset covers 5 different locations (see table 6), two in Canada (Alberta and Ontario) and three in the U.S. (PGAE, SCE, and SDGE). The three locations in the U.S. correspond to three sub-regions of California wholesale market. Locations in the other markets are not included due to lack of data on emission intensity. The time range covers from February 1st through April 22nd of 2011, which corresponds to the time range of the data we have on emission intensity (see section 6.3). The reference time zone is Eastern Time (UTC-05).

For Ontario location, electricity prices are based on Hourly Ontario Energy Price plus Hourly Uplift Charge, which accounts for transmission congestion and line loss. For Alberta location, prices are based on Hourly Pool Price plus Hourly Uplift Charge. Canadian dollars are converted to U.S. dollars at a fixed

[2] Based on 2008 statistics (IRC, 2009)

exchange rate of 0.97093 (2010 average exchange rate[3]). For California locations, their hourly LMP values are used[4].

Table 6: Locations under Study			
Location	Market	Time Zone	Data Source
Ontario	Ontario (IESO)	UTC-05	(IESO, 2011a)
Alberta	Alberta (AESO)	UTC-07	(AESO, 2011c)
PGAE	California (CAISO)	UTC-08	(CAISO, 2011c)
SCE	California (CAISO)	UTC-08	(CAISO, 2011c)
SDGE	California (CAISO)	UTC-08	(CAISO, 2011c)

Figure 31 shows our historical data on electricity price. Prices exhibit highly volatile behavior with frequent outliners, especially at Alberta and Ontario locations. Interestingly prices can go negative, often during the night when power supply exceeds demand. For data centers, this means they could get paid for consuming electricity during these hours.

Figure 32 provides a closer look at daily variations. As expected, prices are lower during the night. Price differentials change from one hour to another. The implication is that dynamic workload distribution solutions will likely outperform static solutions in exploiting these differentials.

[3] http://www.bankofcanada.ca/stats/assets/pdf/nraa-2010.pdf

[4] California price records for 02-Feb-2011 were missing. We interpolated them using the records of the previous and next days (i.e. 01-Feb-2011 & 03-Feb-2011).

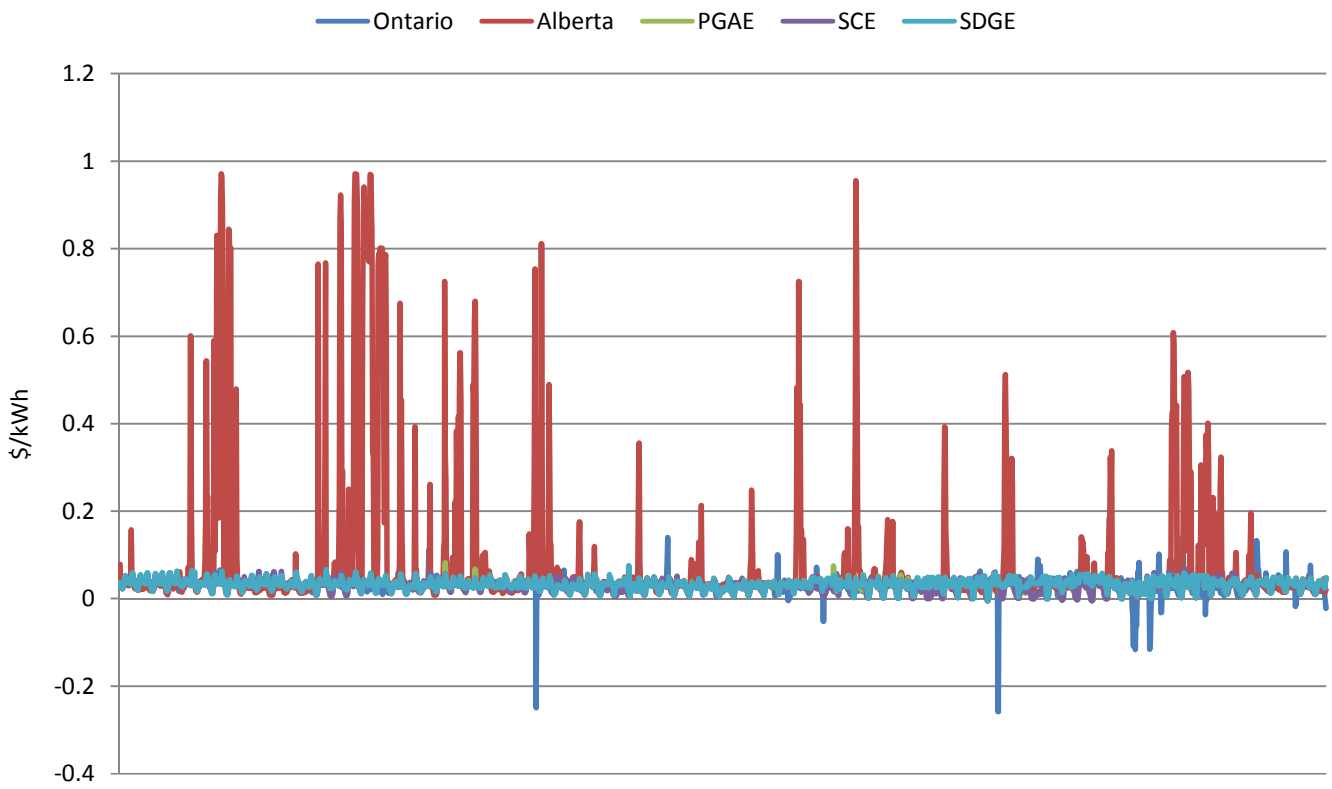


Figure 31: 81-Day Historical Data on Electricity Price (Feb 01 – April 22)

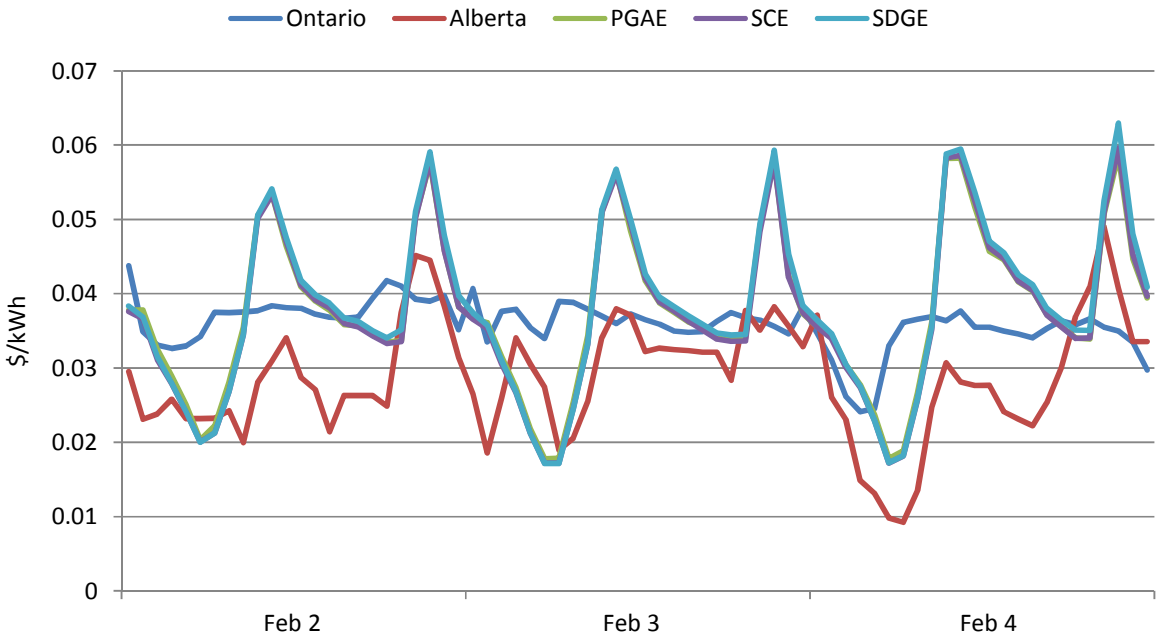


Figure 32: 3-Day Historical Data on Electricity Price (Feb 02 – Feb 04)

6.3 Emission Intensity

Not all kilowatt-hours are created equally; some carry a large footprint than others. Thus, to determine the carbon cost for executing a workload at a data center, it is not sufficient to only measure the amount of energy being consumed. We also need to take into account the emission intensity of the generation mix used to produce that energy.

Electricity generation is typically handled by a mix of base load, intermediate load, and peaking load power plants (Farhat & Ugursal, 2010). The purpose of having a mix of power plants is to satisfy the variability of the demand while minimizing the cost of generation.

- Base load power plants are designed to operate at full capacity on a continuous basis, and have high fixed costs and low operational costs. Commonly, coal-fired, nuclear and hydro power plants are used as base load plants.
- Intermediate load power plants have moderate fixed costs and their operational costs are higher than those of base load units. These power plants run during the daytime, filling the gap in supply between base load and peak load power. They can change their output more easily than base load power plants. Commonly oil, combined cycle natural gas and hydro plants are used as intermediate load power plants.
- Peaking load power plants have lower capital costs and higher operating cost. They are only brought online at peak demand periods since they are the most expensive plants to operate. These plants can change their output easily and quickly to match fluctuations in demand. They can also be started up and shut down quickly. Peaking power plants commonly use natural gas or oil (in combustion turbine or steam cycle plants), and if available, hydro or pumped hydro.

In addition, there are non-dispatchable power plants such as wind, solar, and tidal power plants. These plants cannot be counted on to produce when the need arises since their output depends on the availability of their energy sources. The electricity they generate is usually fed to the grid whenever it becomes available.

The emission intensity of a generation mix depends on the types of fuels included in the mix and how much of each type contributing to the mix. Coal is the most common fuel for generating electricity in North America due to its abundance and low cost. However, coal-fired power plants are the most polluting plants, emitting far more greenhouse gases (GHGs) than their renewable counterparts (see

table 7). Renewable sources such as solar power and wind power produce much less greenhouse gases but their availability varies from region to region and throughout the day and seasons (CEA, 2006).

Table 7: Lifecycle Estimates for Electricity Generators[5]	
Technology	Estimate (gCO_2e/kWh)
Hydroelectric	10
Wind	10
Biogas	11
Solar thermal	13
Biomass (wood waste)	31
Solar PV	32
Geothermal	38
Nuclear	66
Natural gas	443
Fuel cell	664
Diesel	778
Heavy oil	778
Coal (no scrubbing)	1050

[5] Extracted from (Sovacool, 2008)

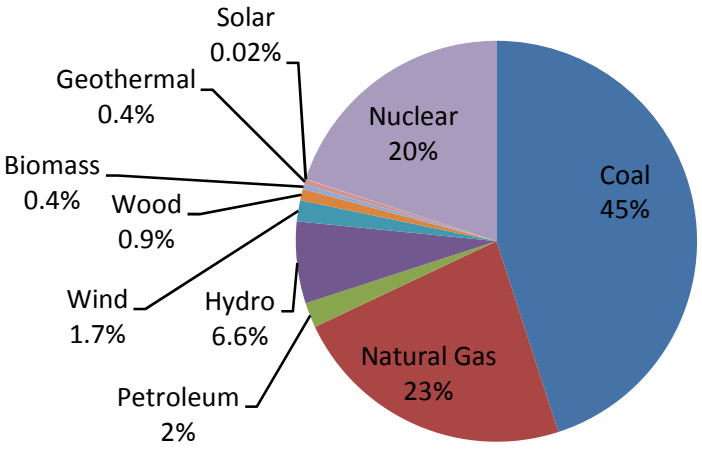


Figure 33: U.S. Electricity Generation by Fuel Type as of 2009 (EIA, 2009)

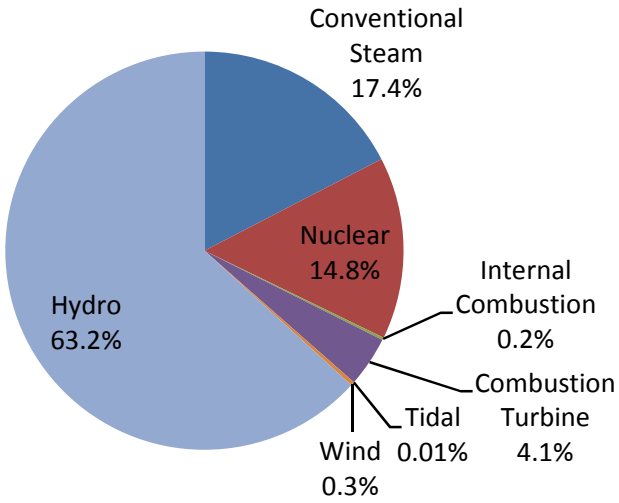


Figure 34: Canada Electricity Generation by Fuel Type as of 2009 (CEA, 2010)

Electricity generation in the United States is dominated by coal (45%), natural gas (23%), and nuclear power (20%) as of 2009. Renewable energy accounts for 10% of total electricity generation (EIA, 2009). In comparison, Canada generates most of its electricity from hydro power (63.2%), conventional steam (17.4%), and nuclear power (14.8%) (CEA, 2010). Individual regions within each country have their own unique generation mixes, depending on the availability of their natural resources and other factors such as transmission line.

Given our records on electricity price, we need to find corresponding records on emission intensity that are in the same time slot and at a comparable level of temporal granularity (i.e. hourly) and geographical granularity (i.e. per price location).

Since emission intensity records that satisfy such requirements do not exist, we derive them from the available data on generation mixes using formula (7) and emission factors from table 5. We calculate the emission intensity of a location as the generation-weighted average of GHG emissions for all electricity generated in the wholesale market region it is associated with. Locations in the same market region are assumed to have the same emission intensity.

For a generation mix with L different fuels, its emission intensity is calculated as follows:

$$\begin{aligned} &Emission\ Intensity\ (gCO_2e/kWh) \\ &= \frac{\sum_{i=1}^{L} Generation\ from\ Fuel\ i\ (kWh) \times GHG\ Emission\ Factor\ for\ Fuel\ i\ (gCO_2e/kWh)}{\sum_{i=1}^{L} Generation\ from\ Fuel\ i\ \ (kWh)} \end{aligned} \quad (7)$$

Ontario. Ontario has a diverse generation mix with nuclear, natural gas, and hydro being the main sources of energy. About 33% of the generating capacity comes from nuclear power, 27% from natural gas, 22.7% from hydro power, 13% from coal, 4% from wind power, and 0.3% from other sources such as wood waste and biogas (IESO, 2011c). Ontario's base load is generated mainly from nuclear and hydro resources, whereas the intermediate and peak loads are supplied by a mix of coal, natural gas, oil and hydroelectric generators with reserve. IESO, the operator of Ontario wholesale market, publishes hourly output for each generating unit over 20 MW that is connected to the transmission grid (IESO, 2011a). The type of fuel that each generator uses is also provided, which allows us to derive the total generation by fuel type and apply formula (7) to calculate the hourly emission intensity for Ontario's mix.

Alberta. Alberta's mix is dominated by thermal sources including coal and natural gas (NG). Coal accounts for 45% of the generating capacity, followed by NG (40%). The remainder comes from hydro power, wind power, and biomass (AESO, 2011b). Alberta's base load is supplied mainly by coal power plants, whereas the intermediate and peak loads are supplied by coal, NG, hydro, and wind power plants. AESO, the operator of Alberta wholesale market, does not publish hourly generator output.

However, they provide hourly records on wind output, internal load (i.e. generation plus import minus export), import/export, and generation outage (i.e. max generating capacity minus available capacity) by fuel type (AESO, 2011c; AESO, 2011d). Given Alberta's rather stable generation mix (dominated by coal and NG) and the above data, we derive the electricity generation by fuel type for each hour as follows:

$$Total\ Gen. = Internal\ Load - Import + Export$$

$$Gen.\ from\ Coal = Max\ Capacity\ of\ Coal - Gen.\ Outage\ of\ Coal$$

$$Gen.\ from\ Hydro = 70\% \times (Max\ Capacity\ of\ Hydro - Gen.\ Outage\ of\ Hydro)$$

$$Gen.\ from\ NG = Total\ Gen. - Gen.\ from\ Wind, Coal, Hydro$$

Thus, we basically subtract total generation by generation from wind and divide the remaining between coal, NG, and hydro under the assumption that coal power plants operate at full capacity (excluding outage), hydro power plants operate at 70% capacity[6] (excluding outage), and NG power plants generate the rest. Having obtained the values for generation by fuel type, we apply formula (7) to calculate the hourly emission intensity for Alberta's mix.

California. California possesses a similar generation mix like Ontario's with the primary energy sources being natural gas, hydro, and nuclear. About 35% of electricity generation comes from natural gas, followed by nuclear (14%), and hydro (12%) (CAISO, 2011b). Unlike Ontario which is a power exporter, California is a power importer, procuring 26% of its power needs from its neighbors. California's base load is handled by nuclear power plants, whereas the intermediate and peak loads are handled by natural gas, hydro, import, geothermal, wind, and other renewable sources. CAISO, the operator of California wholesale market, provides hourly generation output per fuel type (CAISO, 2011a), which we feed to formula (7) to calculate the hourly emission intensity for California' mix.

We obtained a total of 81 days worth of hourly emission intensities for Ontario, Alberta[7], and California. The time range covers from February 1st through April 22nd of 2011.

[6] The data we collect corresponds to the winter season when hydro power plants typically operate below their full capacity. We assume that they operate at 70% of their capacity.

[7] Alberta's records on generation outage per fuel type were not available for April 1-22. Since these were planned generation outages, we assumed a monthly pattern and filled in the missing records using data from the previous month (i.e. March 1-22).

Figure 35 shows our historical data on emission intensity. Compared to electricity price, emission intensity exhibits more predictable behaviors. There is a clear difference between the three locations with Ontario having the lowest emission intensity most of the time, followed by California (i.e. PGAE, SCE, & SDGE) and Alberta. Figure 36 provides a closer look at daily variations.

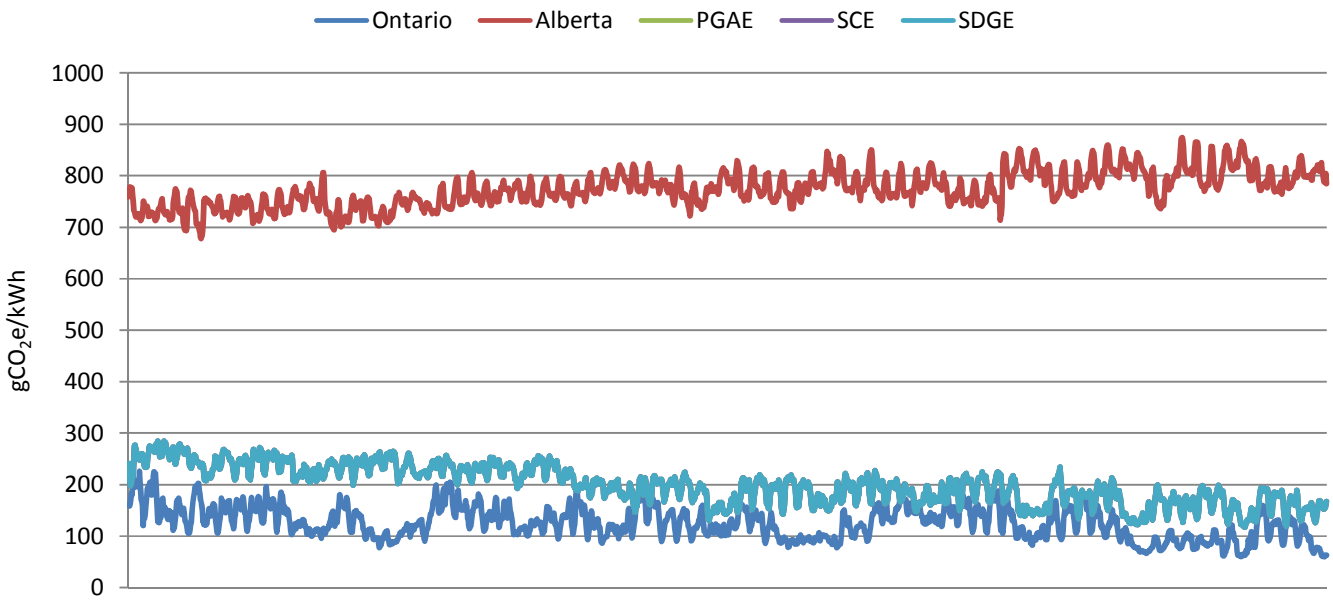


Figure 35: 81-Day Historical Data on Emission Intensity (Feb 01 – April 22)

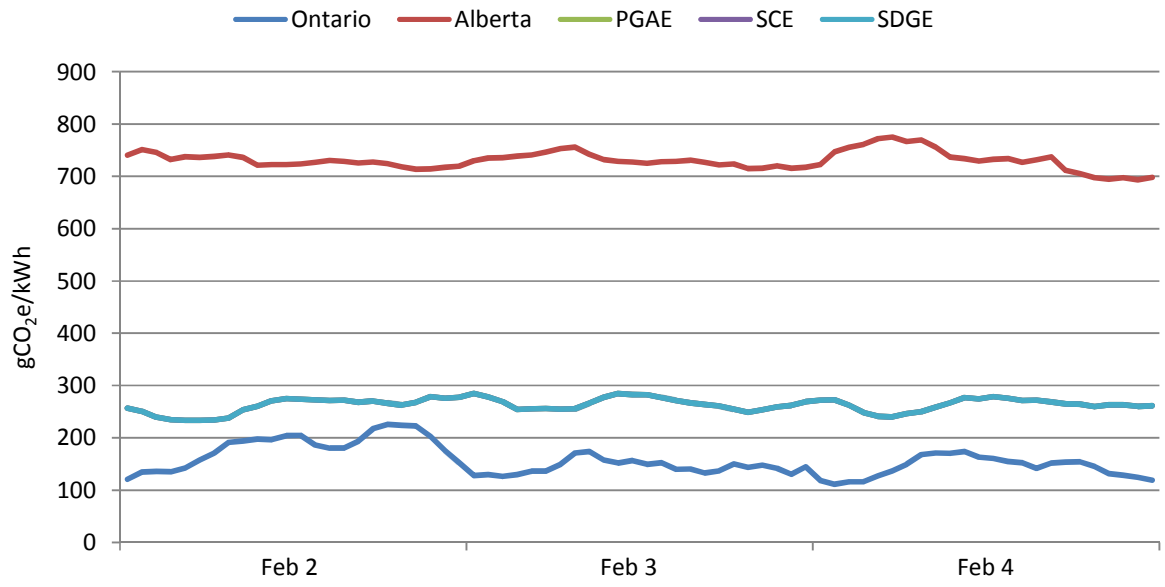


Figure 36: 3-Day Historical Data on Emission Intensity (Feb 02 – Feb 04)

7 Evaluating Optimization Framework

Our study stems from the dual challenge for Cloud Computing—high electricity cost and large carbon footprint. To address this challenge, we propose a workload distribution optimization framework that comprised of three components—a multi-objective optimization model, an aggregate cost function, and six optimization heuristics. We describe our optimization model in section 4.2, our aggregation cost function in section 4.3, and our optimization heuristics in sections 4.4.

Our framework is designed to achieve two primary goals—providing Cloud providers with different trade-off options and reducing their costs through exploiting geographical and temporal variations in external cost factors (i.e. electricity price and emission intensity). We evaluate our framework with respect to these goals by means of simulation.

7.1 Simulation Strategy

We perform two series of simulations. The first series is designed to validate output from our optimization model. We perform validation by examining the workload distribution computed by the Cloud scheduler in each simulation. The second series is designed to evaluate our framework with respect to its design goals, i.e. trade-off handling and cost savings. We evaluate trade-off handing by comparing the average electricity cost and carbon cost per workload/job among optimization heuristics that have different objectives. We evaluate cost savings by comparing the average electricity cost and carbon cost per workload/job between our heuristics and a baseline algorithm that does not exploit geographical and temporal variations in electricity price and emission intensity.

We simulate the optimization process with a computing Cloud that comprises 5 data centers (i.e. $N = 5$ in our optimization model), one at each of the locations under study (see table 6). Figure 37 illustrates the geographical distribution of the data centers. Our simulations range from 12-hour periods to 81-day periods, each divided into 1-hour time slots (i.e. $\delta = 3600$).

Data Center. The data centers have different capacities but feature the same PUE and type of servers. Since we focus on electricity price and emission intensity differentials among data center locations, not the energy efficiency of individual data centers or servers, assuming identical PUE and server type allows us to eliminate biases related to energy usage when comparing the electricity cost and carbon cost among data centers.

Data center characteristics are described in table 8. CPU parameters are adapted from the work in (Sankaranarayanan et al., 2011). PUE is assumed to be 1.3, which is the projected PUE of data centers implementing energy efficiency best practices (EPA, 2007). Data center capacity ranges from hundreds to tens of thousands of CPU hours (i.e. roughly dozens to thousands of 4-CPU servers), corresponding to 3 classes of data centers—localized, mid-tier, and enterprise-class (EPA, 2007).

Workload Generator. The workload generator extracts all the jobs for a given hour from a workload trace and submits them to the Cloud scheduler. We use two types of workload traces: crafted traces (see section 7.2), which we use to validate output from our optimization model, and real-world traces (see section 6.1), which we use to evaluate our framework with respect to its design goals.

Cloud Scheduler. The Cloud scheduler relies on the 6 optimization heuristics described in section 4.4.3 (i.e. GEC, GCF, GB, GECwLA, GCFwLA, & GBwLA) to optimize workload distributions. For each given job, the scheduler allocates it to the data center and time slot that have the least aggregate cost. Aggregate costs are calculated using the aggregate cost function defined in section 3.3.3 and energy usage function defined in section 4.1.3. Jobs in the same time slot are sorted by CPU demand (biggest demand first) and scheduled sequentially in this order. The intuition for this ordering is that since bigger jobs take up more data center capacity, we schedule them first when there is more capacity available. Note that our heuristics are greedy on the aggregate cost for individual jobs; they try to map each job to a data center and time slot that result in the least aggregate cost for that job. Thus, the heuristics don't necessarily produce an optimal distribution of all the jobs in the same time slot. In other words, they don't guarantee a globally optimal solution with regards to the aggregate cost.

24-hour electricity price and emission intensity signals from individual data centers are assumed to be available to the Cloud scheduler (i.e. $K = 24$ in our optimization model). The signals are based on the historical data we collected from electricity market operators (see section 6.2 and 6.3). The carbon price is assumed to be $30 per tonne of CO_2e, which is the targeted carbon tax rate in British Columbia by 2012[8].

[8] http://www.rev.gov.bc.ca/documents_library/notices/British_Columbia_Carbon_Tax.pdf

We implement our optimization framework (i.e. optimization model, aggregate method, & heuristics) and simulator in Java. We use Apache Commons Math package[9] for generating statistics and JAXB library[10] for parsing some of our input files.

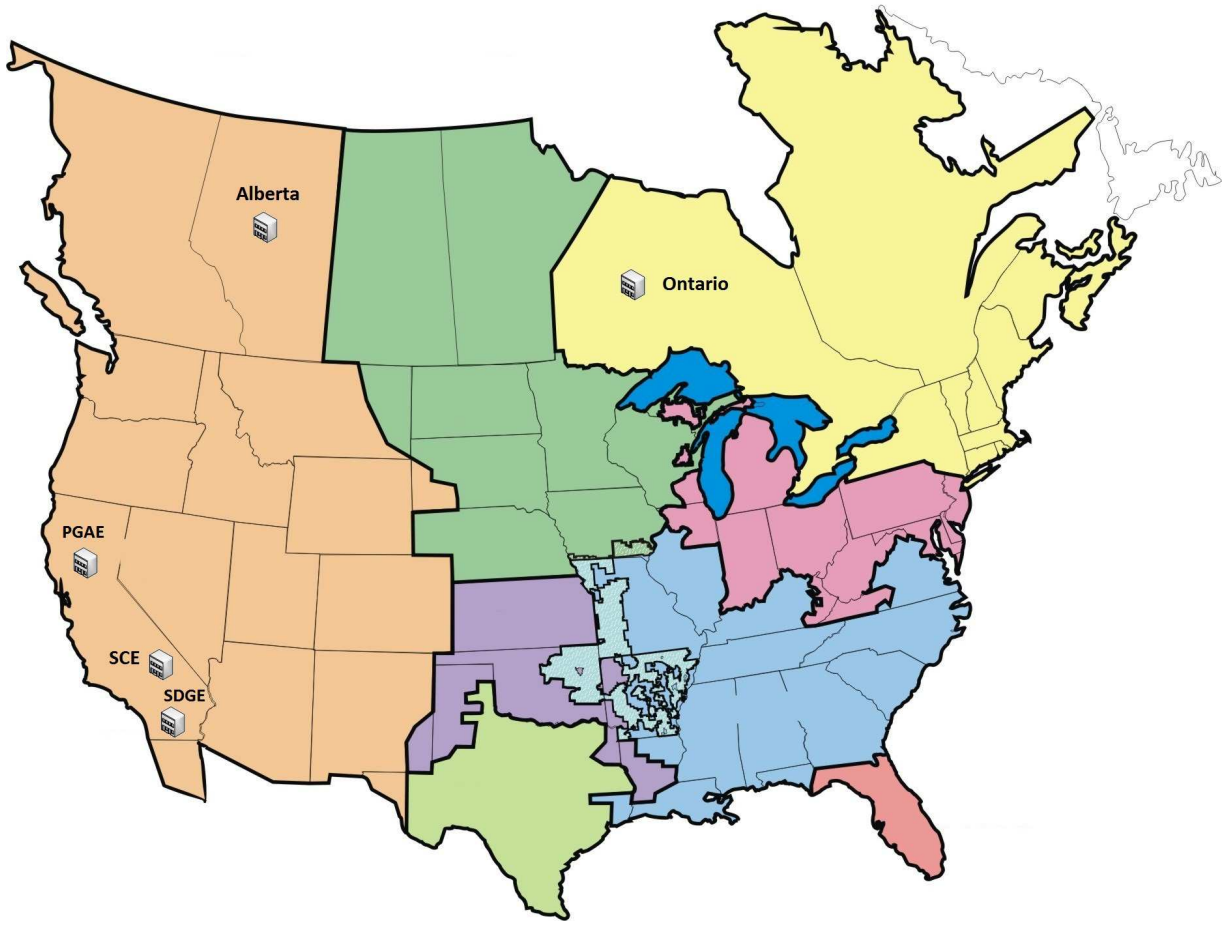


Figure 37: Data Center Locations (based on NERC Regions Map[11])

[9] http://commons.apache.org/math/userguide/stat.html

[10] http://jaxb.java.net/

[11] http://www.nerc.com/fileUploads/File/AboutNERC/maps/NERC_Regions_color.jpg

Table 8: Data Center Characteristics					
Location	CPU Power Factors		CPU Frequency f (GHz)	PUE μ	Max Capacity (Number of CPU Hours)
	β	α			
Ontario	75	5	1.8	1.34	8,000
Alberta	75	5	1.8	1.34	3,600
PGAE	75	5	1.8	1.34	24,000
SCE	75	5	1.8	1.34	1,600
SDGE	75	5	1.8	1.34	200

7.2 Validating Model Output

To determine whether our model works the way it is intended, we run simulations with crafted 12-hour workload traces and examine the resulted workload distributions. Each simulation runs one of our optimization heuristics. Jobs are distributed among 3 data centers—Ontario, Alberta, and PGAE. 24-hour electricity price and emission intensity signals are based on a 2-day worth of records extracted from our historical data on electricity price and emission intensity (see Figure 38 and Figure 39).

The simulations are divided into 3 groups; each has a specific validation objective.

- Same Load, Same Deadline: The objective of these simulations is to determine whether the model steers jobs towards data centers and time slots that have the least (aggregate) costs. We use a 12-hour workload trace that has one job for each hour; each job requires 1000 CPU hours and has a 24-hour deadline (see table 9a).
- Different Loads, Same Deadline: The objective of these simulations is to determine whether the model steers jobs towards the least-cost data centers and time slots without violating capacity constraints. We use a 12-hour workload trace that has 3 jobs for each hour; each job requires a different number of CPU hours but has the same 24-hour deadline (see table 9c). The number of CPU hours required for the 3 jobs is chosen such that no individual data centers can accommodate all of them.

- Different Loads, Different Deadlines: The objective of these simulations is to determine whether the model steers jobs towards the least-cost data centers and time slots without violating capacity and deadline constraints. The workload trace is similar to the previous one but here each job is randomly assigned a deadline in the range 1 to 24 (see table 9e).

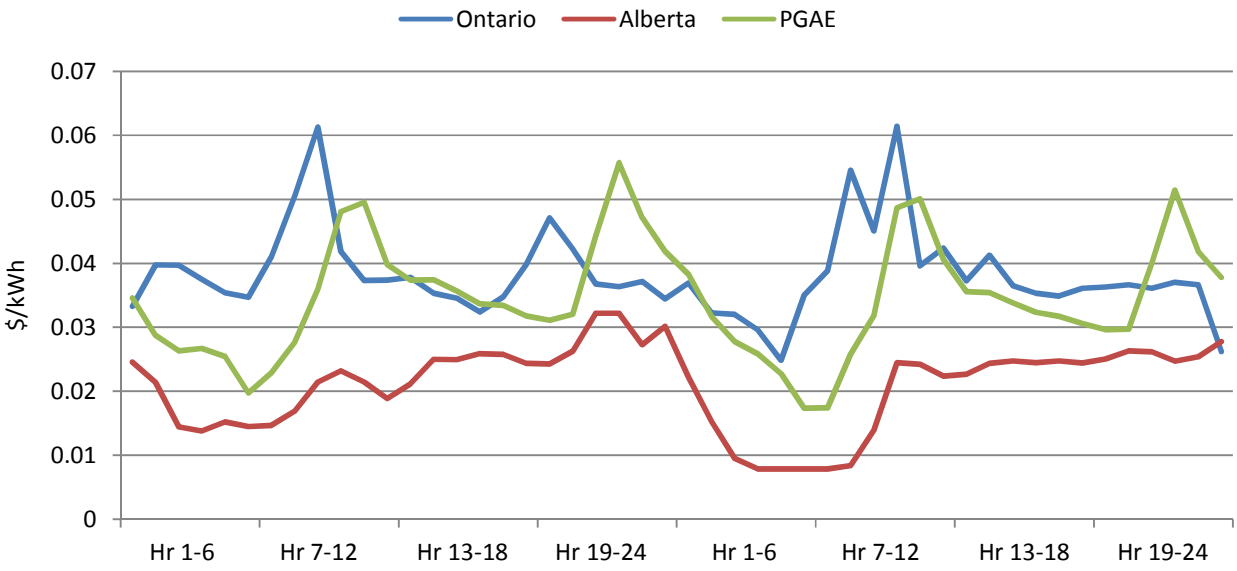


Figure 38: Hourly Electricity Prices, February 10-11, Hour 1 is 12:00-1:00AM

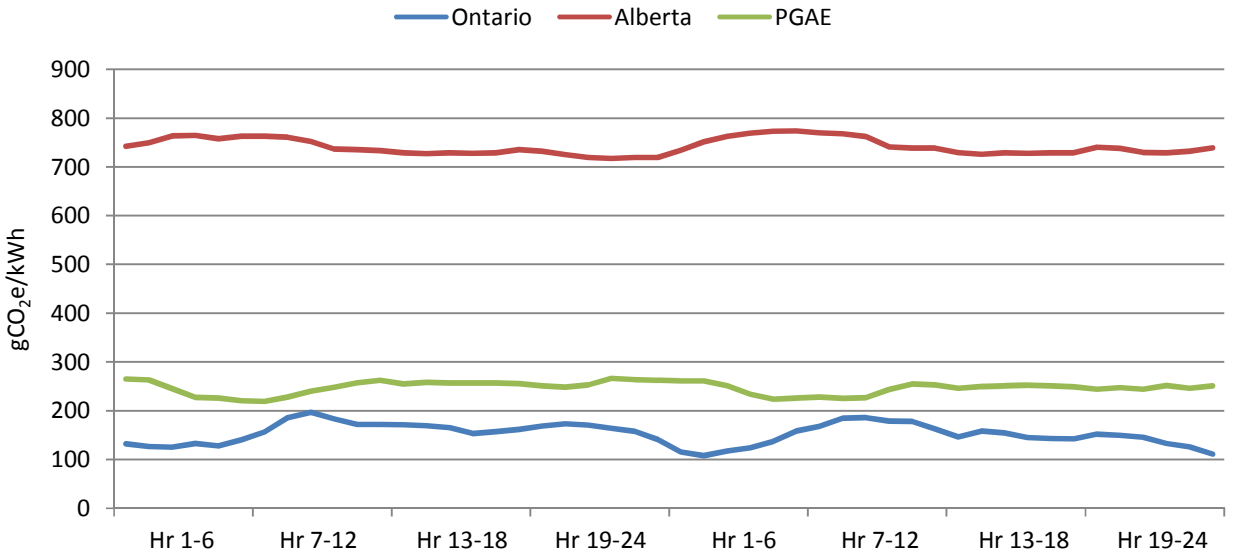


Figure 39: Hourly Emission Intensities, February 10-11, Hour 1 is 12:00-1:00AM

Table 9b, 9d, and 9f show the resulted workload distributions. Overall we find that our model properly follows its specification.

- Jobs are steered towards the least-cost data centers and time slots. For example, in table 9b, job 1 under GEC is scheduled at Alberta, which has the least electricity price within hour 1 of February 10. Under GECwLA, however, job 1 is pushed to hour 4, which has the least electricity price within its 24-hour window.
- Capacity constraints are enforced. For example, in table 9d, job 1a under GCF is scheduled at PGAE even though Ontario has the least emission intensity. Similarly in table 9f, job 1a under GECwLA is scheduled at PGAE even though Alberta has a lower electricity price. The reason is because job 1a requires 18,000 CPU hours, which exceeds Ontario capacity (8,000 CPU hours).
- Deadline constraints are enforced. For example, in table 9d, job 1c under GECwLA is scheduled at Alberta, hour 4 of February 10 even though the electricity price at this location is even lower in hour 1 of February 11. The reason is because hour 1 of February is beyond the 24-hour deadline of job 1c. Similarly, in table 9f, job 2c under GECwLA is scheduled at hour 2 of February 10 even though this time slot does not have the least electricity price within 24-hour window. The reason is because job 2c has a 1-hour deadline, which makes the immediate time slot (hour 2) the only slot that can be considered.

Same Workload, Same Deadline

Table 9a: Input Workload Trace (Same Workload, Same Deadline)

Job Id	Arrival Time Slot	CPU Hours	Deadline
1	Feb10-Hr1	1000	24
2	Feb10-Hr2	1000	24
3	Feb10-Hr3	1000	24
4	Feb10-Hr4	1000	24
5	Feb10-Hr5	1000	24
6	Feb10-Hr6	1000	24
7	Feb10-Hr7	1000	24
8	Feb10-Hr8	1000	24
9	Feb10-Hr9	1000	24
10	Feb10-Hr10	1000	24
11	Feb10-Hr11	1000	24
12	Feb10-Hr12	1000	24

Table 9b: Output Workload Distribution (Same Workload, Same Deadline)												
Job Id	GEC		GCF		GB		GECwLA		GCFwLA		GBwLA	
	Data Center	Execution Execution	Data Center	Execution Time Slot	Data Center	Execution Time Slot	Data Center	Execution Time Slot	Data Center	Execution Time Slot	Data Center	Execution Time Slot
1	Alberta	Feb10-Hr1	Ontario	Feb10-Hr1	Ontario	Feb10-Hr1	Alberta	Feb10-Hr4	Ontario	Feb10-Hr3	PGAE	Feb10-Hr6
2	Alberta	Feb10-Hr2	Ontario	Feb10-Hr2	PGAE	Feb10-Hr2	Alberta	Feb10-Hr4	Ontario	Feb11-Hr1	PGAE	Feb10-Hr6
3	Alberta	Feb10-Hr3	Ontario	Feb10-Hr3	PGAE	Feb10-Hr3	Alberta	Feb10-Hr4	Ontario	Feb11-Hr2	PGAE	Feb10-Hr6
4	Alberta	Feb10-Hr4	Ontario	Feb10-Hr4	PGAE	Feb10-Hr4	Alberta	Feb11-Hr3	Ontario	Feb11-Hr2	PGAE	Feb10-Hr6
5	Alberta	Feb10-Hr5	Ontario	Feb10-Hr5	PGAE	Feb10-Hr5	Alberta	Feb11-Hr4	Ontario	Feb11-Hr2	PGAE	Feb10-Hr6
6	Alberta	Feb10-Hr6	Ontario	Feb10-Hr6	PGAE	Feb10-Hr6	Alberta	Feb11-Hr4	Ontario	Feb11-Hr2	PGAE	Feb10-Hr6
7	Alberta	Feb10-Hr7	Ontario	Feb10-Hr7	PGAE	Feb10-Hr7	Alberta	Feb11-Hr4	Ontario	Feb11-Hr2	PGAE	Feb11-Hr6
8	Alberta	Feb10-Hr8	Ontario	Feb10-Hr8	PGAE	Feb10-Hr8	Alberta	Feb11-Hr5	Ontario	Feb11-Hr2	PGAE	Feb11-Hr6
9	Alberta	Feb10-Hr9	Ontario	Feb10-Hr9	PGAE	Feb10-Hr9	Alberta	Feb11-Hr5	Ontario	Feb11-Hr2	PGAE	Feb11-Hr6
10	Alberta	Feb10-Hr10	Ontario	Feb10-Hr10	Alberta	Feb10-Hr10	Alberta	Feb11-Hr5	Ontario	Feb11-Hr2	PGAE	Feb11-Hr6
11	Alberta	Feb10-Hr11	Ontario	Feb10-Hr11	Ontario	Feb10-Hr11	Alberta	Feb11-Hr6	Ontario	Feb11-Hr1	PGAE	Feb11-Hr6
12	Alberta	Feb10-Hr12	Ontario	Feb10-Hr12	Alberta	Feb10-Hr12	Alberta	Feb11-Hr6	Ontario	Feb11-Hr1	PGAE	Feb11-Hr6

Different Workloads, Same Deadline

Table 9c: Input Workload Trace (Different Workloads, Same Deadline)			
Job Id	Arrival Time Slot	CPU Hours	Deadline
1a	Feb10-Hr1	18000	24
1b	Feb10-Hr1	5000	24
1c	Feb10-Hr1	2000	24
2a	Feb10-Hr2	18000	24
2b	Feb10-Hr2	5000	24
2c	Feb10-Hr2	2000	24
3a	Feb10-Hr2	18000	24
3b	Feb10-Hr3	5000	24
3c	Feb10-Hr3	2000	24
4a	Feb10-Hr3	18000	24
4b	Feb10-Hr4	5000	24
4c	Feb10-Hr4	2000	24
5a	Feb10-Hr4	18000	24

5b	Feb10-Hr5	5000	24
5c	Feb10-Hr5	2000	24
6a	Feb10-Hr5	18000	24
6b	Feb10-Hr6	5000	24
6c	Feb10-Hr6	2000	24
7a	Feb10-Hr6	18000	24
7b	Feb10-Hr7	5000	24
7c	Feb10-Hr7	2000	24
8a	Feb10-Hr7	18000	24
8b	Feb10-Hr8	5000	24
8c	Feb10-Hr8	2000	24
9a	Feb10-Hr8	18000	24
9b	Feb10-Hr9	5000	24
9c	Feb10-Hr9	2000	24
10a	Feb10-Hr9	18000	24
10b	Feb10-Hr10	5000	24
10c	Feb10-Hr10	2000	24
11a	Feb10-Hr10	18000	24
11b	Feb10-Hr11	5000	24
11c	Feb10-Hr11	2000	24
12a	Feb10-Hr11	18000	24
12b	Feb10-Hr12	5000	24
12c	Feb10-Hr12	2000	24

Table 9d: Output Workload Distribution (Different Workloads, Same Deadline)

Job Id	GEC		GCF		GB		GECwLA		GCFwLA		GBwLA	
	Data Center	Execution Time Slot	Data Center	Execution Time Slot	Data Center	Execution Time Slot	Data Center	Execution Time Slot	Data Center	Execution Time Slot	Data Center	Execution Time Slot
1a	PGAE	Feb10-Hr1	PGAE	Feb10-Hr1	PGAE	Feb10-Hr1	PGAE	Feb10-Hr6	PGAE	Feb10-Hr7	PGAE	Feb10-Hr6
1b	Ontario	Feb10-Hr1	Ontario	Feb10-Hr1	Ontario	Feb10-Hr1	PGAE	Feb10-Hr6	Ontario	Feb10-Hr3	PGAE	Feb10-Hr6
1c	Alberta	Feb10-Hr1	Ontario	Feb10-Hr1	Ontario	Feb10-Hr1	Alberta	Feb10-Hr4	Ontario	Feb10-Hr3	PGAE	Feb10-Hr7
2a	PGAE	Feb10-Hr2	PGAE	Feb10-Hr2	PGAE	Feb10-Hr2	PGAE	Feb10-Hr7	PGAE	Feb10-Hr6	PGAE	Feb10-Hr7
2b	PGAE	Feb10-Hr2	Ontario	Feb10-Hr2	PGAE	Feb10-Hr2	PGAE	Feb10-Hr7	Ontario	Feb11-Hr1	PGAE	Feb10-Hr5
2c	Alberta	Feb10-Hr2	Ontario	Feb10-Hr2	Ontario	Feb10-Hr2	Alberta	Feb10-Hr3	Ontario	Feb11-Hr1	PGAE	Feb10-Hr7
3a	PGAE	Feb10-Hr3	PGAE	Feb10-Hr3	PGAE	Feb10-Hr3	PGAE	Feb10-Hr5	PGAE	Feb10-Hr5	PGAE	Feb10-Hr5

3b	PGAE	Feb10-Hr3	Ontario	Feb10-Hr3	PGAE	Feb10-Hr3	PGAE	Feb10-Hr5	Ontario	Feb11-Hr2	PGAE	Feb10-Hr4
3c	Alberta	Feb10-Hr3	Ontario	Feb10-Hr3	Alberta	Feb10-Hr3	Alberta	Feb10-Hr6	Ontario	Feb11-Hr2	PGAE	Feb10-Hr7
4a	PGAE	Feb10-Hr4	PGAE	Feb10-Hr4	PGAE	Feb10-Hr4	PGAE	Feb10-Hr4	PGAE	Feb10-Hr4	PGAE	Feb10-Hr4
4b	PGAE	Feb10-Hr4	Ontario	Feb10-Hr4	PGAE	Feb10-Hr4	PGAE	Feb10-Hr4	Ontario	Feb11-Hr3	PGAE	Feb10-Hr8
4c	Alberta	Feb10-Hr4	Ontario	Feb10-Hr4	Alberta	Feb10-Hr4	Alberta	Feb11-Hr3	Ontario	Feb11-Hr3	Alberta	Feb11-Hr3
5a	PGAE	Feb10-Hr5	PGAE	Feb10-Hr5	PGAE	Feb10-Hr5	PGAE	Feb11-Hr4	PGAE	Feb10-Hr8	PGAE	Feb11-Hr4
5b	PGAE	Feb10-Hr5	Ontario	Feb10-Hr5	PGAE	Feb10-Hr5	PGAE	Feb11-Hr4	Ontario	Feb11-Hr4	PGAE	Feb11-Hr4
5c	Alberta	Feb10-Hr5	Ontario	Feb10-Hr5	Alberta	Feb10-Hr5	Alberta	Feb11-Hr4	Ontario	Feb11-Hr4	Alberta	Feb11-Hr4
6a	PGAE	Feb10-Hr6	PGAE	Feb10-Hr6	PGAE	Feb10-Hr6	PGAE	Feb11-Hr5	PGAE	Feb11-Hr5	PGAE	Feb11-Hr5
6b	PGAE	Feb10-Hr6	Ontario	Feb10-Hr6	PGAE	Feb10-Hr6	PGAE	Feb11-Hr5	Ontario	Feb11-Hr5	Ontario	Feb11-Hr5
6c	Alberta	Feb10-Hr6	Ontario	Feb10-Hr6	Alberta	Feb10-Hr6	Alberta	Feb11-Hr5	Ontario	Feb11-Hr5	Ontario	Feb11-Hr5
7a	PGAE	Feb10-Hr7	PGAE	Feb10-Hr7	PGAE	Feb10-Hr7	PGAE	Feb11-Hr6	PGAE	Feb11-Hr6	PGAE	Feb11-Hr6
7b	PGAE	Feb10-Hr7	Ontario	Feb10-Hr7	PGAE	Feb10-Hr7	PGAE	Feb11-Hr6	Ontario	Feb10-Hr24	PGAE	Feb11-Hr6
7c	Alberta	Feb10-Hr7	Ontario	Feb10-Hr7	Alberta	Feb10-Hr7	Alberta	Feb11-Hr6	Ontario	Feb10-Hr24	PGAE	Feb11-Hr5
8a	PGAE	Feb10-Hr8	PGAE	Feb10-Hr8	PGAE	Feb10-Hr8	PGAE	Feb11-Hr7	PGAE	Feb11-Hr7	PGAE	Feb11-Hr7
8b	PGAE	Feb10-Hr8	Ontario	Feb10-Hr8	PGAE	Feb10-Hr8	PGAE	Feb11-Hr7	Ontario	Feb10-Hr16	PGAE	Feb11-Hr7
8c	Alberta	Feb10-Hr8	Ontario	Feb10-Hr8	Alberta	Feb10-Hr8	Alberta	Feb11-Hr7	Ontario	Feb10-Hr16	PGAE	Feb11-Hr5
9a	PGAE	Feb10-Hr9	PGAE	Feb10-Hr9	PGAE	Feb10-Hr9	PGAE	Feb11-Hr8	PGAE	Feb11-Hr8	PGAE	Feb11-Hr8
9b	PGAE	Feb10-Hr9	Ontario	Feb10-Hr9	PGAE	Feb10-Hr9	Ontario	Feb11-Hr5	Ontario	Feb10-Hr17	PGAE	Feb11-Hr8
9c	Alberta	Feb10-Hr9	Ontario	Feb10-Hr9	Alberta	Feb10-Hr9	Alberta	Feb11-Hr8	Ontario	Feb10-Hr17	PGAE	Feb11-Hr5
10a	PGAE	Feb10-Hr10	PGAE	Feb10-Hr10	PGAE	Feb10-Hr10	PGAE	Feb11-Hr3	PGAE	Feb11-Hr9	PGAE	Feb11-Hr3
10b	Ontario	Feb10-Hr10	Ontario	Feb10-Hr10	Ontario	Feb10-Hr10	PGAE	Feb11-Hr8	Ontario	Feb10-Hr23	Ontario	Feb11-Hr4
10c	Alberta	Feb10-Hr10	Ontario	Feb10-Hr10	Alberta	Feb10-Hr10	Alberta	Feb11-Hr9	Ontario	Feb10-Hr23	Alberta	Feb11-Hr7
11a	PGAE	Feb10-Hr11	PGAE	Feb10-Hr11	PGAE	Feb10-Hr11	PGAE	Feb10-Hr19	PGAE	Feb11-Hr4	PGAE	Feb10-Hr19
11b	Ontario	Feb10-Hr11	Ontario	Feb10-Hr11	Ontario	Feb10-Hr11	PGAE	Feb11-Hr3	Ontario	Feb11-Hr6	PGAE	Feb11-Hr3
11c	Alberta	Feb10-Hr11	Ontario	Feb10-Hr11	Ontario	Feb10-Hr11	Alberta	Feb11-Hr2	Ontario	Feb11-Hr6	Alberta	Feb11-Hr5
12a	PGAE	Feb10-Hr12	PGAE	Feb10-Hr12	PGAE	Feb10-Hr12	PGAE	Feb11-Hr2	PGAE	Feb11-Hr10	PGAE	Feb11-Hr9
12b	Ontario	Feb10-Hr12	Ontario	Feb10-Hr12	Ontario	Feb10-Hr12	Ontario	Feb11-Hr4	Ontario	Feb10-Hr18	Ontario	Feb11-Hr2
12c	Alberta	Feb10-Hr12	Ontario	Feb10-Hr12	Alberta	Feb10-Hr12	Alberta	Feb10-Hr12	Ontario	Feb10-Hr18	Alberta	Feb11-Hr6

Different Workloads, Different Deadlines

Table 9e: Input Workload Trace (Different Workloads, Different Deadlines)

Job Id	Arrival Time Slot	CPU Hours	Deadline
1a	Feb10-Hr1	18000	7

1b	Feb10-Hr1	5000	14
1c	Feb10-Hr1	2000	24
2a	Feb10-Hr2	18000	4
2b	Feb10-Hr2	5000	3
2c	Feb10-Hr2	2000	1
3a	Feb10-Hr2	18000	8
3b	Feb10-Hr3	5000	2
3c	Feb10-Hr3	2000	17
4a	Feb10-Hr3	18000	11
4b	Feb10-Hr4	5000	19
4c	Feb10-Hr4	2000	7
5a	Feb10-Hr4	18000	5
5b	Feb10-Hr5	5000	4
5c	Feb10-Hr5	2000	13
6a	Feb10-Hr5	18000	15
6b	Feb10-Hr6	5000	12
6c	Feb10-Hr6	2000	19
7a	Feb10-Hr6	18000	10
7b	Feb10-Hr7	5000	18
7c	Feb10-Hr7	2000	21
8a	Feb10-Hr7	18000	13
8b	Feb10-Hr8	5000	2
8c	Feb10-Hr8	2000	17
9a	Feb10-Hr8	18000	20
9b	Feb10-Hr9	5000	21
9c	Feb10-Hr9	2000	6
10a	Feb10-Hr9	18000	9
10b	Feb10-Hr10	5000	20
10c	Feb10-Hr10	2000	24
11a	Feb10-Hr10	18000	19
11b	Feb10-Hr11	5000	23
11c	Feb10-Hr11	2000	13
12a	Feb10-Hr11	18000	21
12b	Feb10-Hr12	5000	17
12c	Feb10-Hr12	2000	24

Table 9f: Output Workload Distribution (Different Workloads, Different Deadlines)												
Job Id	GEC		GCF		GB		GECwLA		GCFwLA		GBwLA	
	Data Center	Execution Time Slot	Data Center	Execution Time Slot	Data Center	Execution Time Slot	Data Center	Execution Time Slot	Data Center	Execution Time Slot	Data Center	Execution Time Slot
1a	PGAE	Feb10-Hr1	PGAE	Feb10-Hr1	PGAE	Feb10-Hr1	PGAE	Feb10-Hr6	PGAE	Feb10-Hr7	PGAE	Feb10-Hr6
1b	Ontario	Feb10-Hr1	Ontario	Feb10-Hr1	Ontario	Feb10-Hr1	PGAE	Feb10-Hr6	Ontario	Feb10-Hr3	PGAE	Feb10-Hr6
1c	Alberta	Feb10-Hr1	Ontario	Feb10-Hr1	Ontario	Feb10-Hr1	Alberta	Feb10-Hr4	Ontario	Feb10-Hr3	PGAE	Feb10-Hr7
2a	PGAE	Feb10-Hr2	PGAE	Feb10-Hr2	PGAE	Feb10-Hr2	PGAE	Feb10-Hr5	PGAE	Feb10-Hr5	PGAE	Feb10-Hr5
2b	PGAE	Feb10-Hr2	Ontario	Feb10-Hr2	PGAE	Feb10-Hr2	PGAE	Feb10-Hr3	Ontario	Feb10-Hr2	PGAE	Feb10-Hr4
2c	Alberta	Feb10-Hr2	Ontario	Feb10-Hr2	Ontario	Feb10-Hr2	Alberta	Feb10-Hr2	Ontario	Feb10-Hr2	PGAE	Feb10-Hr2
3a	PGAE	Feb10-Hr3	PGAE	Feb10-Hr3	PGAE	Feb10-Hr3	PGAE	Feb10-Hr7	PGAE	Feb10-Hr6	PGAE	Feb10-Hr7
3b	PGAE	Feb10-Hr3	Ontario	Feb10-Hr3	PGAE	Feb10-Hr3	PGAE	Feb10-Hr3	Ontario	Feb10-Hr4	PGAE	Feb10-Hr4
3c	Alberta	Feb10-Hr3	Ontario	Feb10-Hr3	Alberta	Feb10-Hr3	Alberta	Feb10-Hr3	Ontario	Feb10-Hr5	PGAE	Feb10-Hr7
4a	PGAE	Feb10-Hr4	PGAE	Feb10-Hr4	PGAE	Feb10-Hr4	PGAE	Feb10-Hr4	PGAE	Feb10-Hr4	PGAE	Feb10-Hr8
4b	PGAE	Feb10-Hr4	Ontario	Feb10-Hr4	PGAE	Feb10-Hr4	PGAE	Feb10-Hr7	Ontario	Feb10-Hr5	PGAE	Feb10-Hr5
4c	Alberta	Feb10-Hr4	Ontario	Feb10-Hr4	Alberta	Feb10-Hr4	Alberta	Feb10-Hr6	Ontario	Feb10-Hr4	PGAE	Feb10-Hr7
5a	PGAE	Feb10-Hr5	PGAE	Feb10-Hr5	PGAE	Feb10-Hr5	PGAE	Feb10-Hr8	PGAE	Feb10-Hr8	PGAE	Feb10-Hr9
5b	PGAE	Feb10-Hr5	Ontario	Feb10-Hr5	PGAE	Feb10-Hr5	PGAE	Feb10-Hr5	Ontario	Feb10-Hr6	PGAE	Feb10-Hr8
5c	Alberta	Feb10-Hr5	Ontario	Feb10-Hr5	Alberta	Feb10-Hr5	Alberta	Feb10-Hr7	Ontario	Feb10-Hr6	Ontario	Feb10-Hr16
6a	PGAE	Feb10-Hr6	PGAE	Feb10-Hr6	PGAE	Feb10-Hr6	PGAE	Feb10-Hr19	PGAE	Feb10-Hr9	PGAE	Feb10-Hr19
6b	PGAE	Feb10-Hr6	Ontario	Feb10-Hr6	PGAE	Feb10-Hr6	PGAE	Feb10-Hr8	Ontario	Feb10-Hr16	Ontario	Feb10-Hr16
6c	Alberta	Feb10-Hr6	Ontario	Feb10-Hr6	Alberta	Feb10-Hr6	Alberta	Feb10-Hr8	Ontario	Feb10-Hr24	Alberta	Feb10-Hr6
7a	PGAE	Feb10-Hr7	PGAE	Feb10-Hr7	PGAE	Feb10-Hr7	PGAE	Feb10-Hr16	PGAE	Feb10-Hr10	PGAE	Feb10-Hr16
7b	PGAE	Feb10-Hr7	Ontario	Feb10-Hr7	PGAE	Feb10-Hr7	PGAE	Feb10-Hr19	Ontario	Feb10-Hr24	PGAE	Feb10-Hr19
7c	Alberta	Feb10-Hr7	Ontario	Feb10-Hr7	Alberta	Feb10-Hr7	Alberta	Feb11-Hr3	Ontario	Feb11-Hr2	Alberta	Feb11-Hr3
8a	PGAE	Feb10-Hr8	PGAE	Feb10-Hr8	PGAE	Feb10-Hr8	PGAE	Feb10-Hr18	PGAE	Feb10-Hr20	PGAE	Feb10-Hr18
8b	PGAE	Feb10-Hr8	Ontario	Feb10-Hr8	PGAE	Feb10-Hr8	PGAE	Feb10-Hr9	Ontario	Feb10-Hr8	PGAE	Feb10-Hr9
8c	Alberta	Feb10-Hr8	Ontario	Feb10-Hr8	Alberta	Feb10-Hr8	Alberta	Feb10-Hr12	Ontario	Feb10-Hr16	Ontario	Feb10-Hr24
9a	PGAE	Feb10-Hr9	PGAE	Feb10-Hr9	PGAE	Feb10-Hr9	PGAE	Feb11-Hr4	PGAE	Feb11-Hr4	PGAE	Feb11-Hr4
9b	PGAE	Feb10-Hr9	Ontario	Feb10-Hr9	PGAE	Feb10-Hr9	PGAE	Feb11-Hr5	Ontario	Feb11-Hr2	Ontario	Feb11-Hr5
9c	Alberta	Feb10-Hr9	Ontario	Feb10-Hr9	Alberta	Feb10-Hr9	Alberta	Feb10-Hr13	Ontario	Feb10-Hr14	Ontario	Feb10-Hr14
10a	PGAE	Feb10-Hr10	PGAE	Feb10-Hr10	PGAE	Feb10-Hr10	PGAE	Feb10-Hr17	PGAE	Feb10-Hr13	PGAE	Feb10-Hr17
10b	Ontario	Feb10-Hr10	Ontario	Feb10-Hr10	Ontario	Feb10-Hr10	PGAE	Feb11-Hr5	Ontario	Feb11-Hr1	PGAE	Feb11-Hr5
10c	Alberta	Feb10-Hr10	Ontario	Feb10-Hr10	Alberta	Feb10-Hr10	Alberta	Feb11-Hr4	Ontario	Feb11-Hr1	PGAE	Feb11-Hr6
11a	PGAE	Feb10-Hr11	PGAE	Feb10-Hr11	PGAE	Feb10-Hr11	PGAE	Feb11-Hr3	PGAE	Feb11-Hr5	PGAE	Feb11-Hr5
11b	Ontario	Feb10-Hr11	Ontario	Feb10-Hr11	Ontario	Feb10-Hr11	PGAE	Feb11-Hr6	Ontario	Feb11-Hr3	PGAE	Feb11-Hr6

11c	Alberta	Feb10-Hr11	Ontario	Feb10-Hr11	Ontario	Feb10-Hr11	Alberta	Feb10-Hr11	Ontario	Feb10-Hr17	PGAE	Feb10-Hr18
12a	PGAE	Feb10-Hr12	PGAE	Feb10-Hr12	PGAE	Feb10-Hr12	PGAE	Feb11-Hr6	PGAE	Feb11-Hr8	PGAE	Feb11-Hr7
12b	Ontario	Feb10-Hr12	Ontario	Feb10-Hr12	Ontario	Feb10-Hr12	PGAE	Feb11-Hr4	Ontario	Feb11-Hr4	PGAE	Feb11-Hr4
12c	Alberta	Feb10-Hr12	Ontario	Feb10-Hr12	Alberta	Feb10-Hr12	Alberta	Feb11-Hr5	Ontario	Feb11-Hr3	PGAE	Feb11-Hr6

7.3 Analyzing Simulation Results

7.3.1 Trade-Off & Savings

We design these simulations around the questions: for a computing Cloud that implements our framework, can our optimization heuristics produce different trade-offs between electricity cost and carbon cost? Considering the workload trace collected from Lawrence Livermore National Lab (see section 6.1), how much electricity cost and carbon cost could have been saved if those computation jobs had been run on a computing Cloud that implemented our framework rather than in the lab?

To answer these questions, we run simulations with the collected workload trace and analyze the average electricity cost and carbon cost per job. There are 5 data centers where jobs can be scheduled—Ontario, Alberta, PGAE, SCE, & SDGE. 24-hour electricity price and emission intensity signals are based on the historical data we collected from electricity market operators (see section 6.2 and 6.3).

To determine whether our framework reduce costs, we need a baseline to compare against. We establish the baseline by running a simulation for a simple algorithm that schedules all jobs at the same data center—PGAE, which is in the same region with Lawrence Livermore National Lab—and in their immediate time slots. All input parameters are the same. Since all jobs are executed in the same data center, there is no shifting of jobs to low-cost locations, hence no exploiting of geographical variations. Also, since there is no delay in job execution, there is no exploiting of temporal variations either.

Figure 40 & 41 show the costs produced by our scheduling algorithms compared to the baseline. Among the heuristics that do not use look-ahead, GEC has the lowest electricity cost and highest carbon cost per job, GCF has the highest electricity cost and lowest carbon cost per job, and GB is somewhat in the middle, having neither highest or lowest costs. It is clear that electricity cost can be reduced at the expense of carbon cost, and vice versa. Also, it is possible to reduce both simultaneously. The results of GECwLA, GCFwLA, & GBwLA show that look-ahead amplifies savings on both costs while maintaining a similar trade-off pattern between them. Thus, Cloud providers can have different trade-offs between

electricity cost and carbon cost by switching from one of our heuristics to another (e.g., GEC to GCF or GECwLA to GCFwLA).

The difference between the highest cost and lowest cost is substantial. For electricity cost, the difference is about 18.74% (i.e. 108.10% – 89.36%) of the baseline when look-ahead is not used and 41.5% when look-ahead is used. For carbon cost, the difference is about 39.76% of the baseline when look-ahead is not used and 42.25% when look-ahead is used. These large cost differentials imply there exists significant trade-off between electricity cost and carbon cost; thus, optimizing solely for electricity cost can result in substantial penalty on carbon cost and vice versa. Our algorithms allow both costs to be reduced simultaneously.

Compared to the baseline, we find that by processing jobs immediately but routing them to data centers with lower costs our framework saves up to 10.64% on electricity cost or 21.25% on carbon cost, or 8.09% on electricity cost and 11.25% on carbon cost simultaneously. By delaying job execution to later time slots with lower costs (i.e. using look-ahead), our framework increases savings up to 53.35% on electricity cost, or 29.13% on carbon cost, or 51.44% on electricity cost and 13.14% on carbon cost simultaneously. Thus, electricity cost and carbon cost do not always compete with each other; by choosing a more balanced workload distribution strategy such as GB and GBwLA, Cloud providers can reduce both costs at the same time.

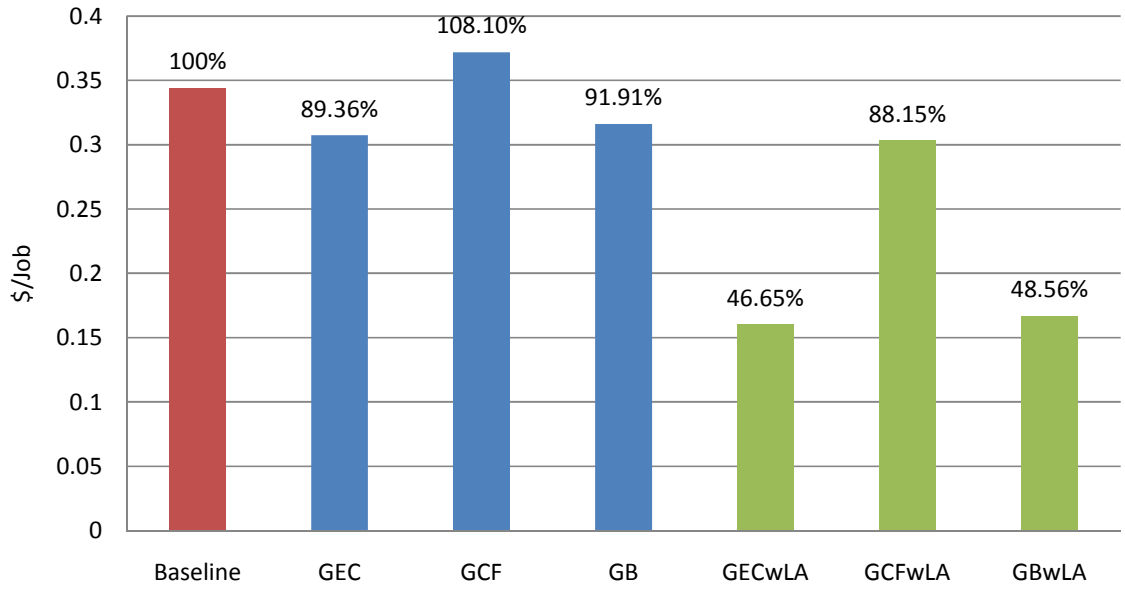


Figure 40: Comparison of Average Electricity Cost per Job among Algorithms

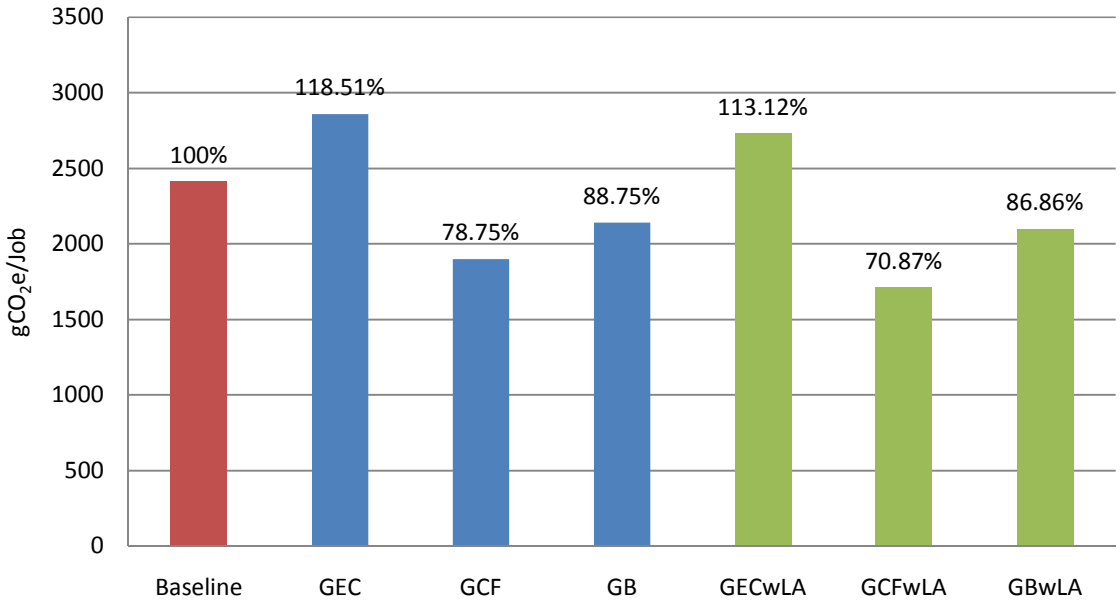


Figure 41: Comparison of Average Carbon Cost per Job among Algorithms

To better understand where the savings come from, we examine the geographical and temporal distributions of workloads produced by each algorithm.

Figure 42 & 43 show the distribution of workloads in time & space by each algorithm. They complement the previous figures by describing exactly where and when workloads were scheduled. For example, by looking at figure 40 & 41, we know that GB which distributes workloads across multiple data centers saves 8.09% of electricity cost and 11.25% of carbon cost compared to the baseline. But what we do not know is at which data centers the workload were scheduled. Figure 42 gives us the answer—most of the workloads were scheduled at Ontario, SCE, & PGAE. Similarly, figure 40 & 41 tell us that GBwLA which employ look-ahead produces significantly more savings than GB. What they do not tell us when the workloads were scheduled. Figure 43 gives us the answer—more than 80% of the workloads were scheduled at a later hour rather than their immediate hour.

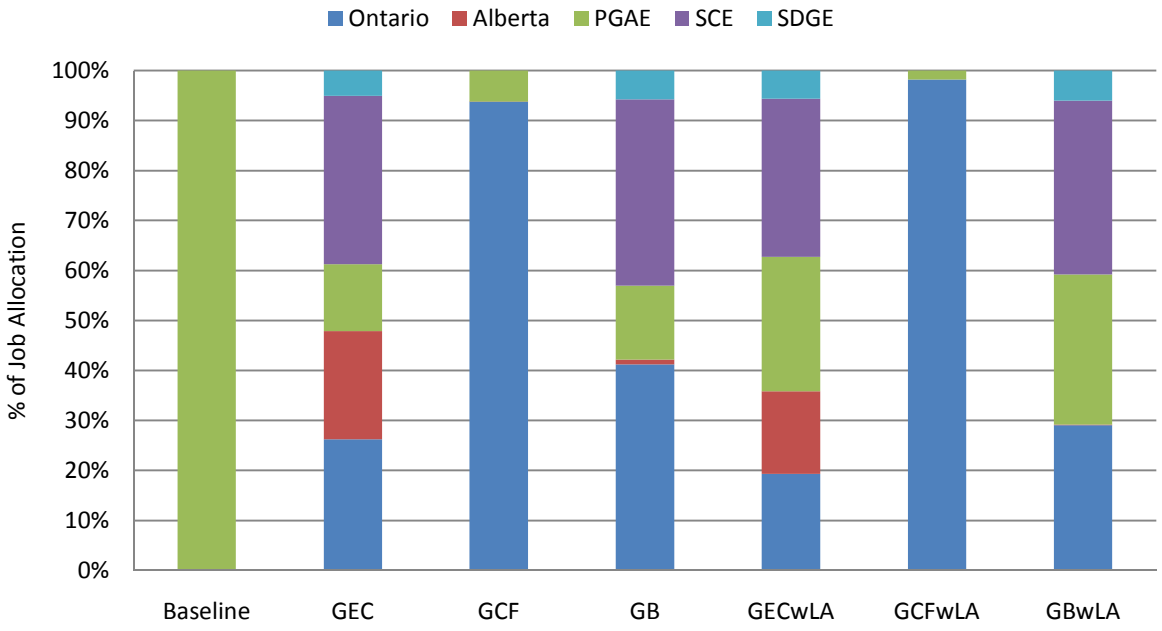


Figure 42: Comparison of Geographical Workload Distribution among Algorithms

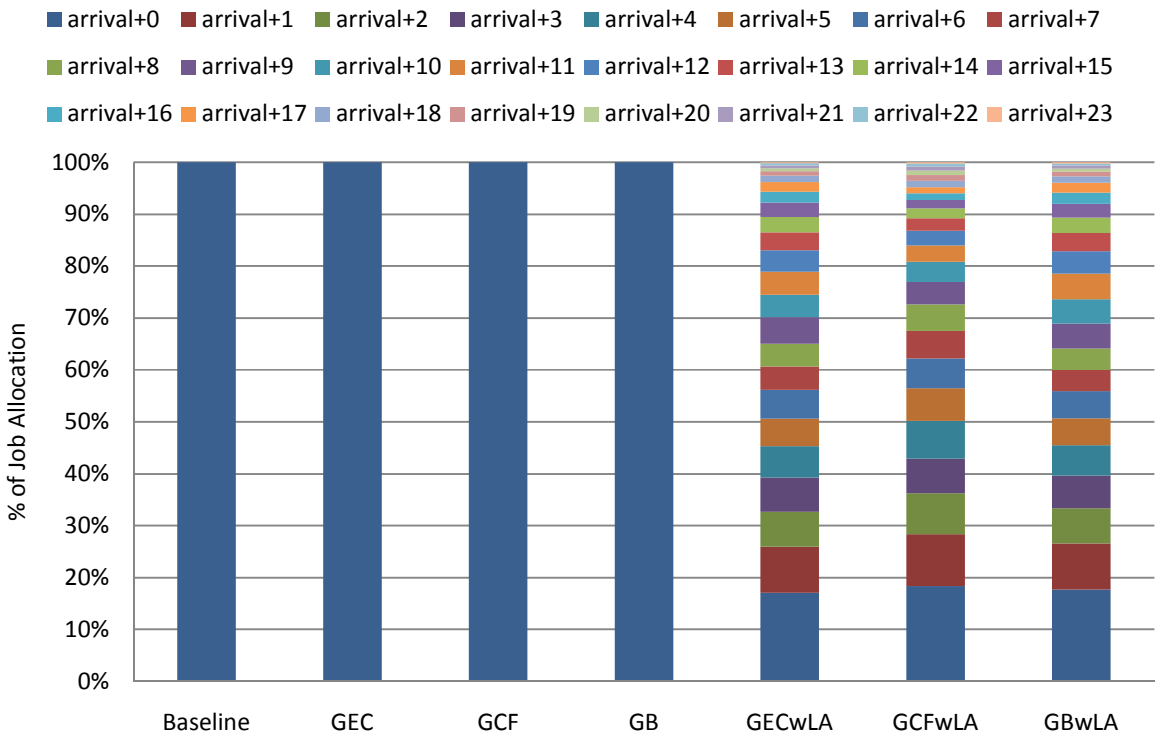


Figure 43: Comparison of Temporal Workload Distribution among Algorithms

7.3.2 Job Deadline

The jump in savings when look-ahead is used suggests that having flexible job deadlines amplifies our ability to exploit cost differentials and hence increases achievable savings. This begs the question: how sensitive are our results to job deadlines?

We run simulations for GECwLA, GCFwLA, & GBwLA with identical input parameters except that all jobs have the same deadline, starting from 1-hour and increasing by 1 up to 24-hour. We choose GECwLA, GCFwLA, & GBwLA because they employ look-ahead, allowing us to assess how changes in job deadline affect scheduling outcomes. As for the other heuristics, they do not use look-ahead, hence changing job deadlines has no impact on their outcomes.

GECwLA

As shown in Figure 44 & 45, costs drop as job deadline gets longer although electricity cost drops faster than carbon cost, suggesting that reduction on electricity cost is more sensitive to changes in deadline than reduction on carbon cost. This result is consistent with what we see in the sample data on electricity price and emission intensity (see section 6.2 & 6.3): the order of data center locations by electricity price often changes from one hour to another whereas their order by emission intensity is relatively stable over time.

The average electricity cost per job under GECwLA drops as much as 70.29% (i.e. 100% – 29.71%) of the baseline when job deadline is set to 24 hours. Almost half of that savings can be achieved with a 6-hour job deadline.

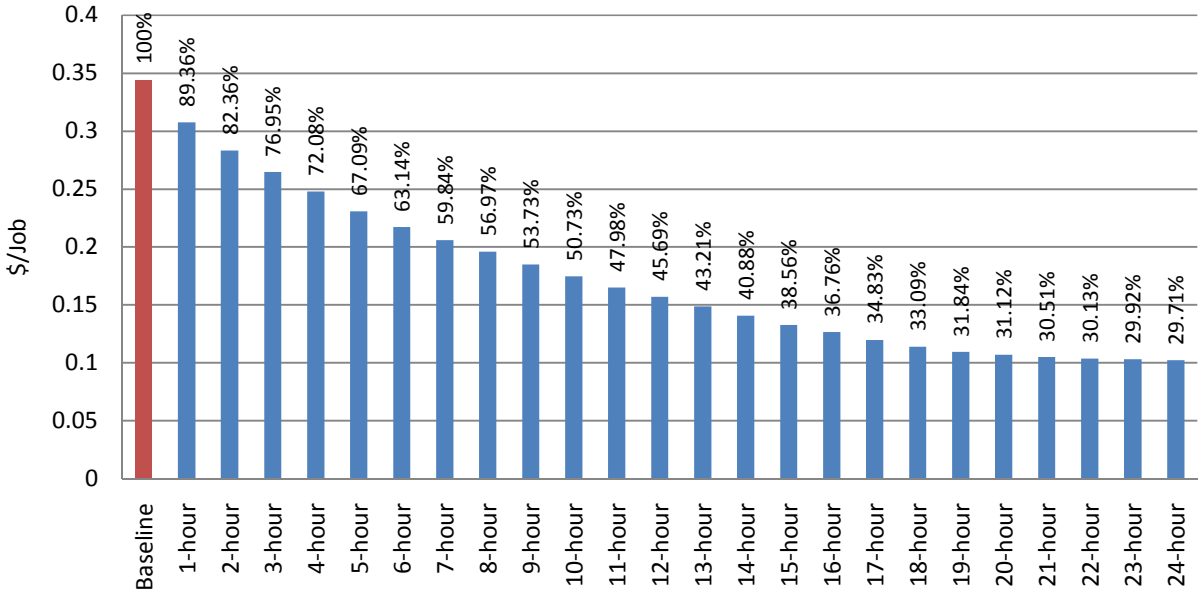


Figure 44: Comparison of Average Electricity Cost per Job between Baseline & GECwLA (Deadline 1-24 Hours)

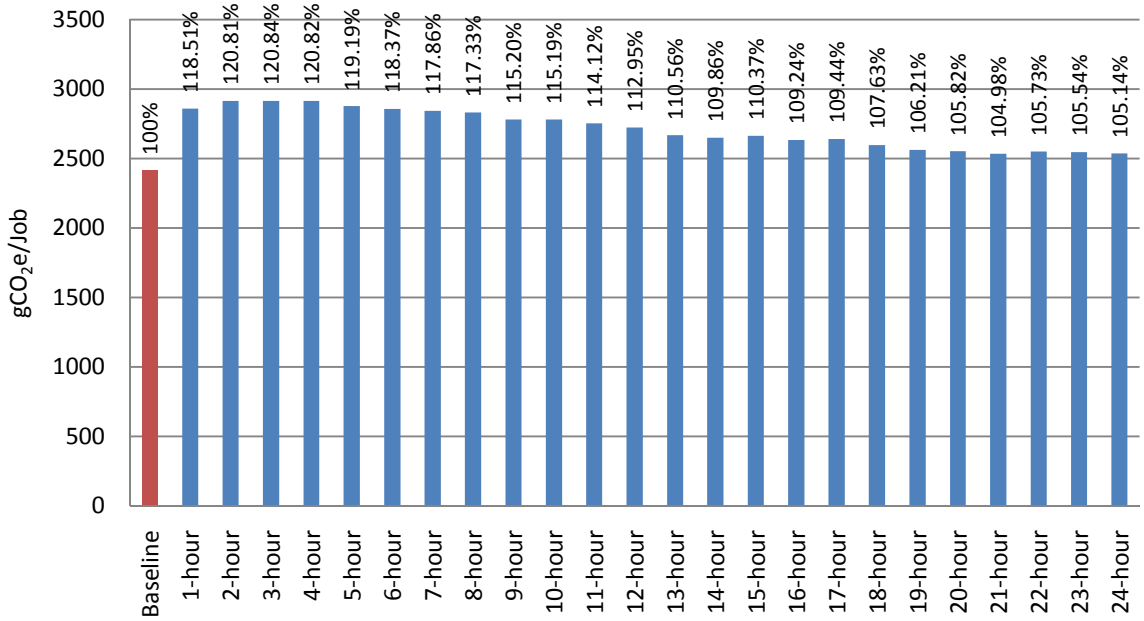


Figure 45: Comparison of Average Carbon Cost per Job between Baseline & GECwLA (Deadline 1-24 Hours)

To better understand where the savings come from, we examine the geographical and temporal distributions of workloads corresponding to each job deadline.

Figure 46 & 47 show how the distributions of workloads in space and time change with job deadline. They complement the previous figures by describing exactly where and when workloads were scheduled. For example, by looking at figure 44, we know that GECwLA saves 23.05% of electricity cost by delaying workload execution by three hour. But what we do not know is where and when the workload were scheduled. Figure 46 & 47 give us the answer—most of the workloads were scheduled at Ontario, SCE, & Alberta. Also, most of the workloads were scheduled at a later hour rather than their immediate hour.

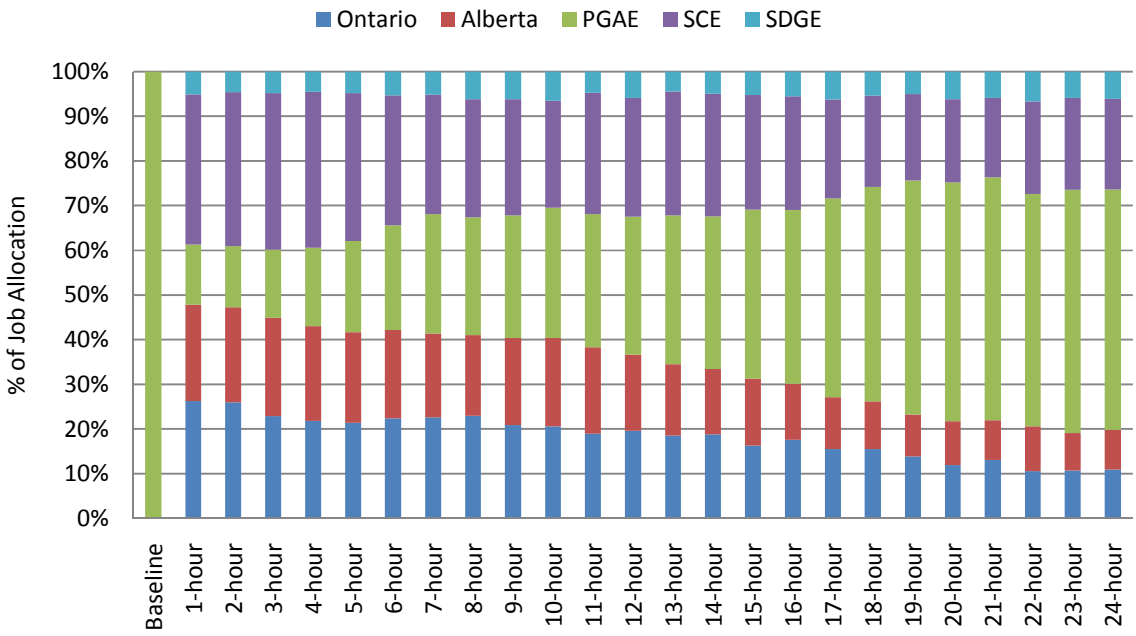


Figure 46: Comparison of Geographical Workload Distribution between Baseline & GECwLA (Deadline 1-24 Hours)

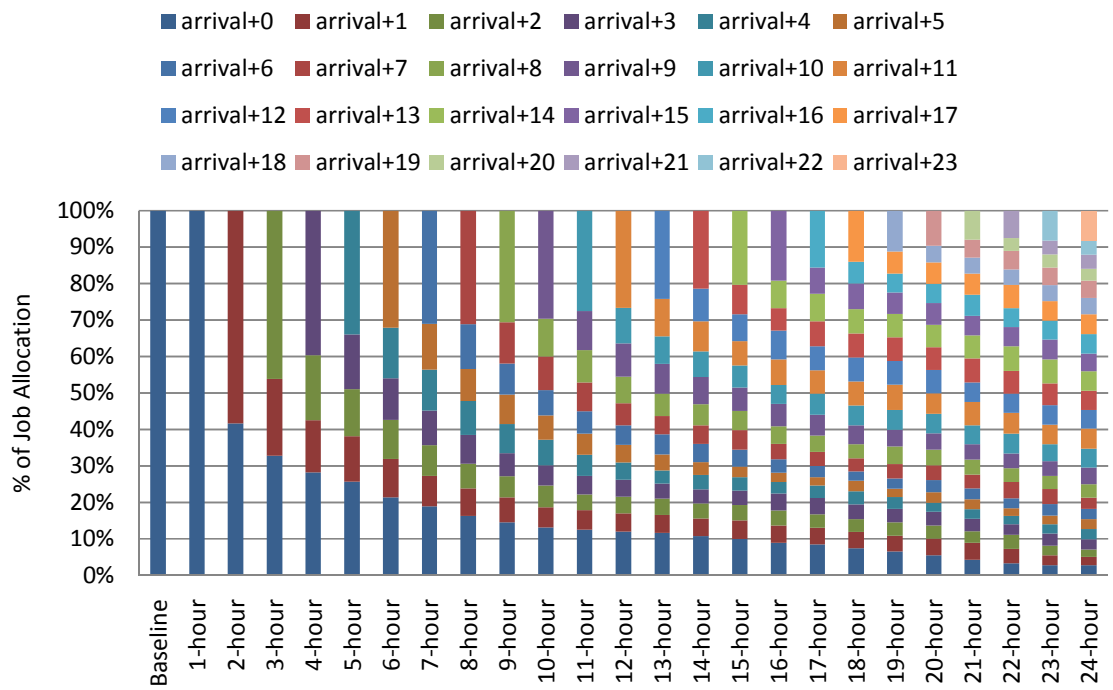


Figure 47: Comparison of Temporal Workload Distribution between Baseline & GECwLA (Deadline 1-24 Hours)

GCFwLA

As shown in Figure 48 & 49, costs drop as job deadline gets longer although electricity cost drops faster than carbon cost, suggesting that reduction on electricity cost is more sensitive to changes in deadline than reduction on carbon cost. This result is consistent with what we see in the sample data on electricity price and emission intensity (see section 6.2 & 6.3): the order of data center locations by electricity price often changes from one hour to another whereas their order by emission intensity is relatively stable over time.

The average carbon cost under GCFwLA drops as much as 33.24% of the baseline when job deadline is set to 24 hours. Almost two third of that savings can be achieved with a 1-hour deadline, suggesting that savings on carbon cost mostly comes from routing jobs to different locations rather than from delaying job execution.

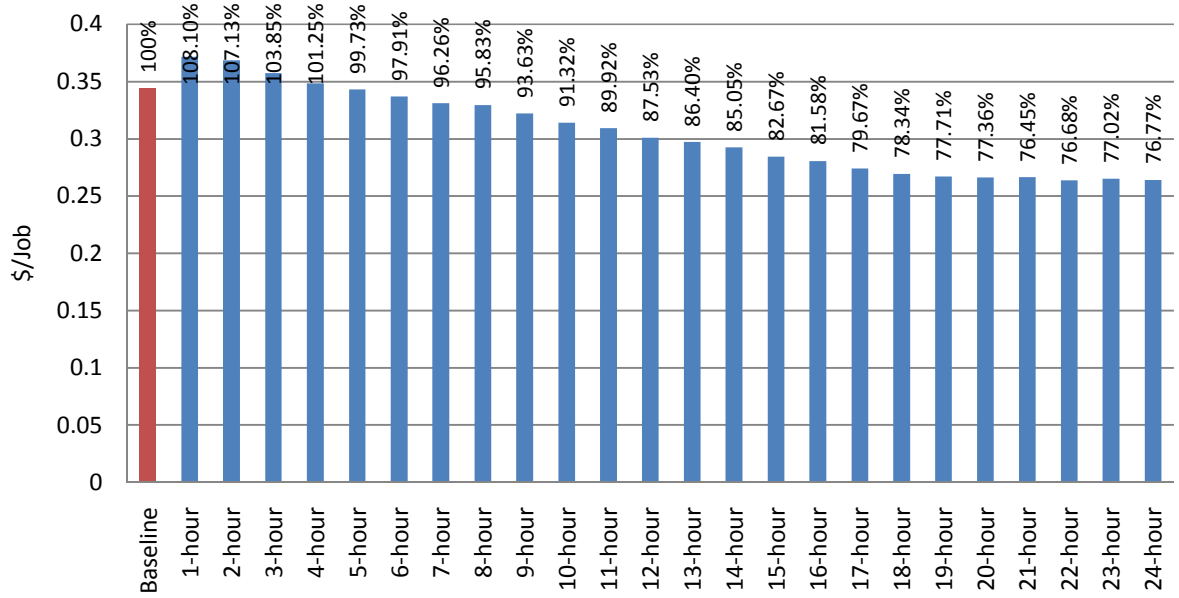


Figure 48: Comparison of Average Electricity Cost per Job between Baseline & GCFwLA (Deadline 1-24 Hours)

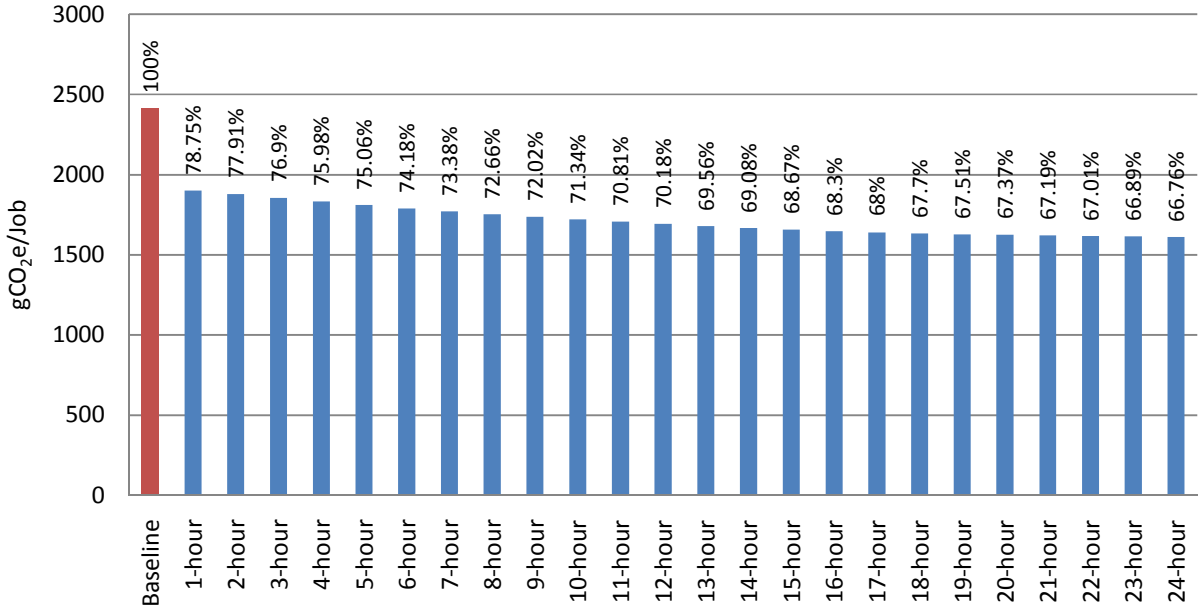


Figure 49: Comparison of Average Carbon Cost per Job between Baseline & GCFwLA (Deadline 1-24 Hours)

To better understand where the savings come from, we examine the geographical and temporal distributions of workloads corresponding to each job deadline.

Figure 50 & 51 show how the distributions of workloads in space and time change with job deadline. They complement the previous figures by describing exactly where and when workloads were scheduled. For example, by looking at figure 49, we know that GCFwLA saves 23.1% of carbon cost by delaying workload execution by three hour. But what we do not know is where and when the workload were scheduled. Figure 50 & 51 give us the answer—most of the workloads were scheduled at Ontario. Also, most of the workloads were scheduled at a later hour rather than their immediate hour.

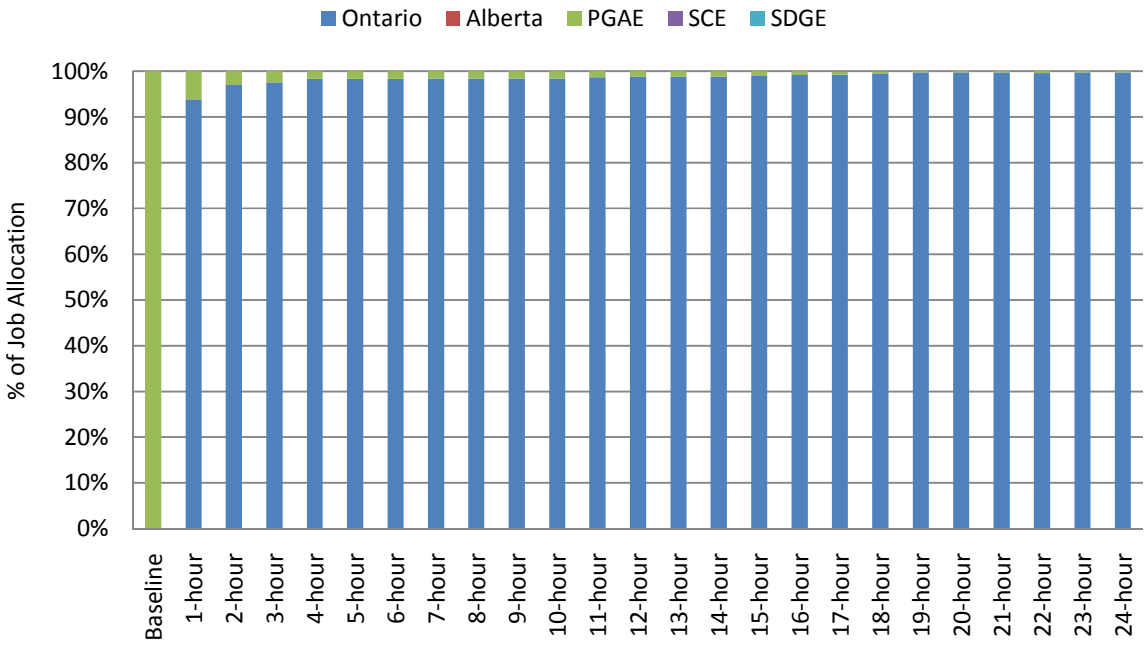


Figure 50: Comparison of Geographical Workload Distribution between Baseline & GCFwLA (Deadline 1-24 Hours)

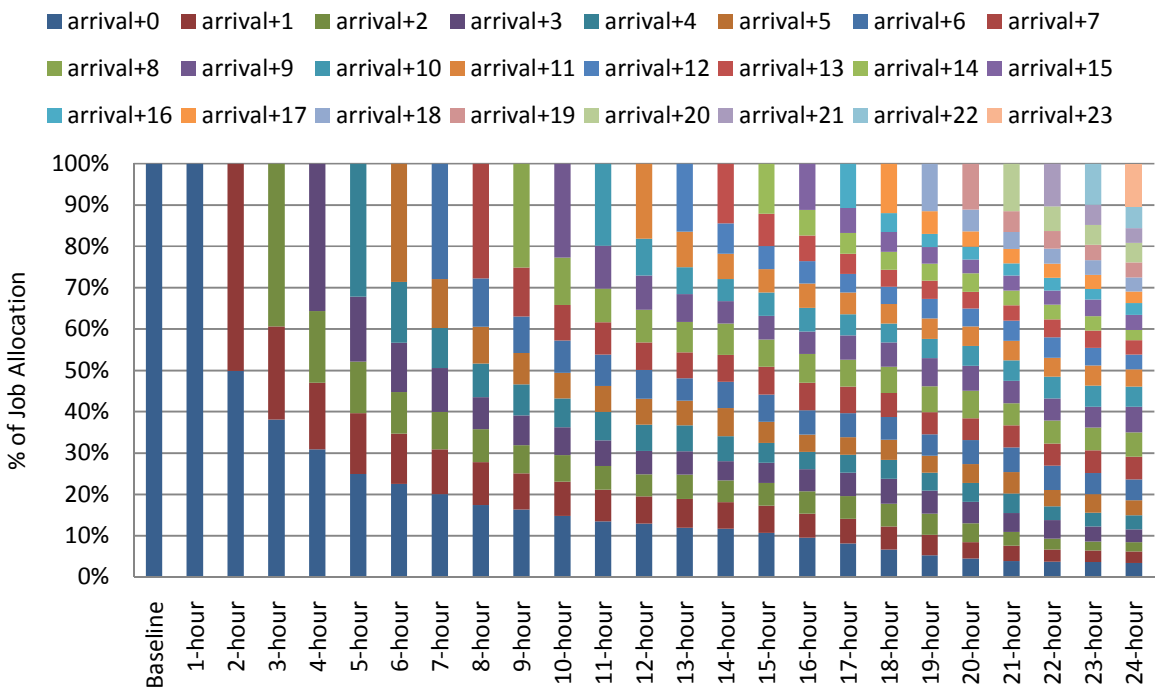


Figure 51: Comparison of Temporal Workload Distribution between Baseline & GCFwLA (Deadline 1-24 Hours)

GBwLA

As shown in Figure 52 & 53, savings generally increases as job deadline gets longer although electricity cost drops faster than carbon cost, suggesting that savings on electricity cost is more sensitive to changes in deadline than savings on carbon cost. This result is consistent with what we see in the sample data on electricity price and emission intensity (see section 6.2 & 6.3): the order of data center locations by electricity price often changes from one hour to another whereas their order by emission intensity is relatively stable over time.

GBwLA continues to produce substantial savings on both costs, amid at lower rates on electricity cost and carbon cost than GECwLA and GCFwLA respectively. Interestingly, while longer deadlines lead to further reduction on electricity cost, it is not always the case for carbon cost. Figure 53 shows that average carbon cost per job rises slightly as job deadline increases from 17 hours to 20 hours whereas average electricity cost per job consistently drops. This suggests electricity cost overwhelms carbon cost in cost aggregation, i.e. savings on the former more than compensates for the penalty on the latter.

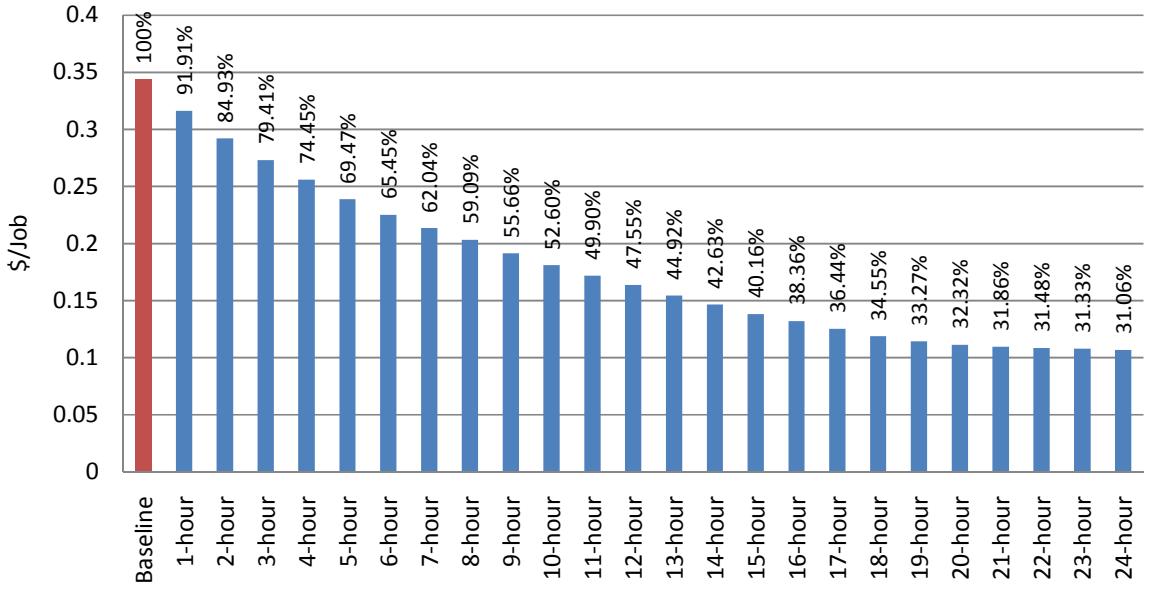


Figure 52: Comparison of Average Electricity Cost per Job between Baseline & GBwLA (Deadline 1-24 Hours)

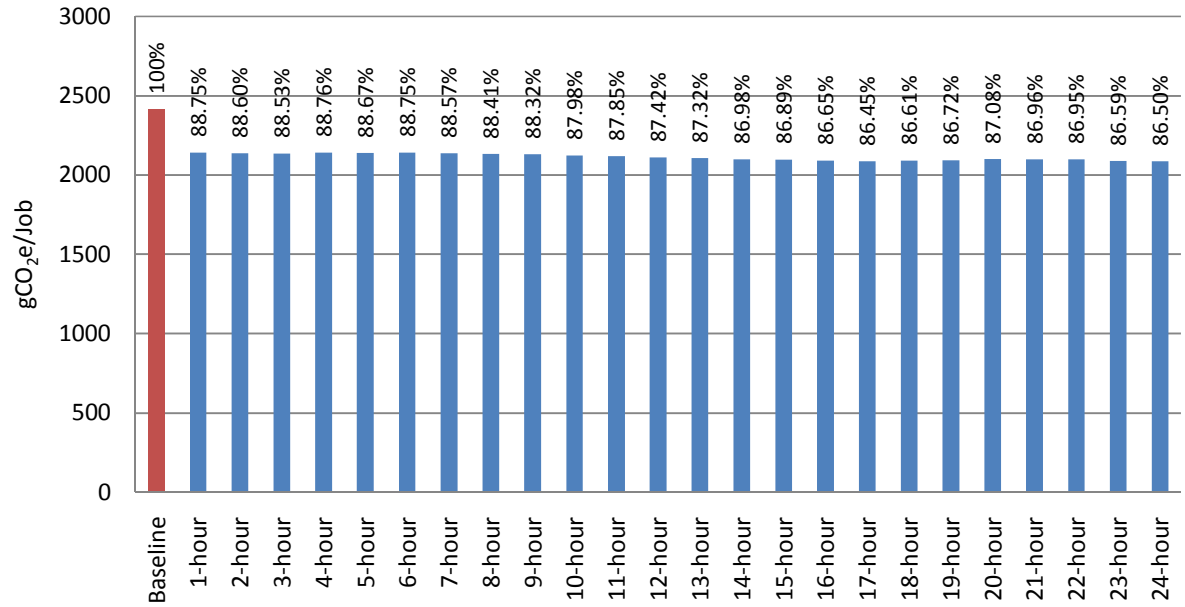


Figure 53: Comparison of Average Carbon Cost per Job between Baseline & GBwLA (Deadline 1-24 Hours)

To better understand where the savings come from, we examine the geographical and temporal distributions of workloads corresponding to each job deadline.

Figure 54 & 55 show how the distributions of workloads in space and time change with job deadline. They complement the previous figures by describing exactly where and when workloads were scheduled. For example, by looking at figure 52 & 53, we know that GBwLA saves 20.59% of electricity cost and 11.47% of carbon cost by delaying workload execution by three hour. But what we do not know is where and when the workload were scheduled. Figure 54 & 55 give us the answer—most of the workloads were scheduled at Ontario, SCE, & PGAE. Also, most of the workloads were scheduled at a later hour rather than their immediate hour.

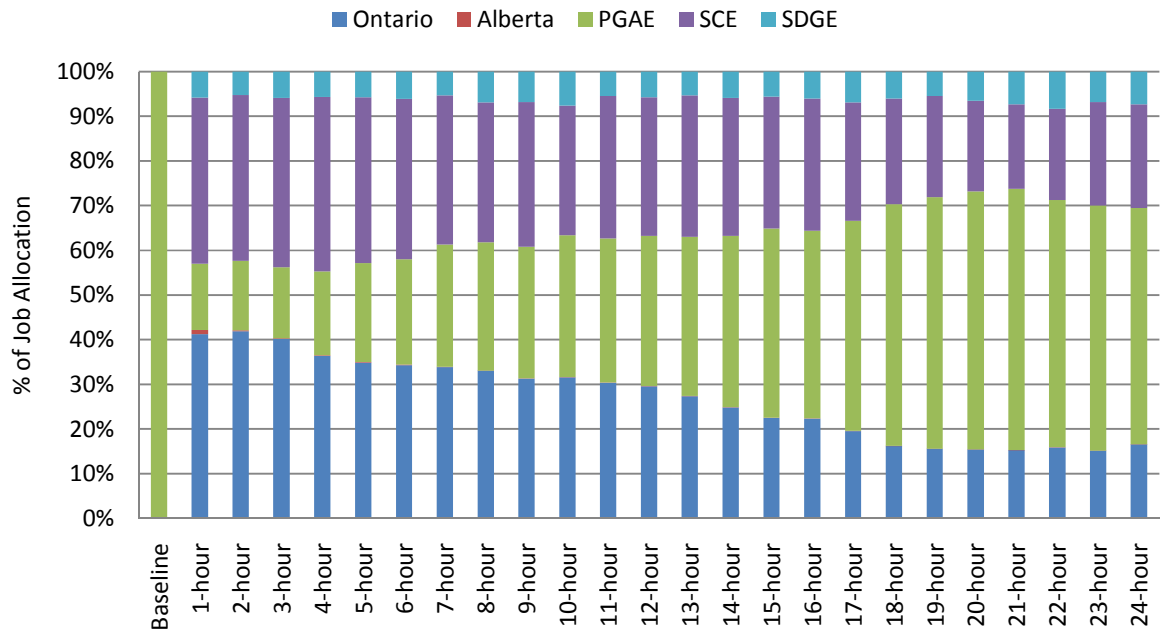


Figure 54: Comparison of Geographical Workload Distribution between Baseline & GBwLA (Deadline 1-24 Hours)

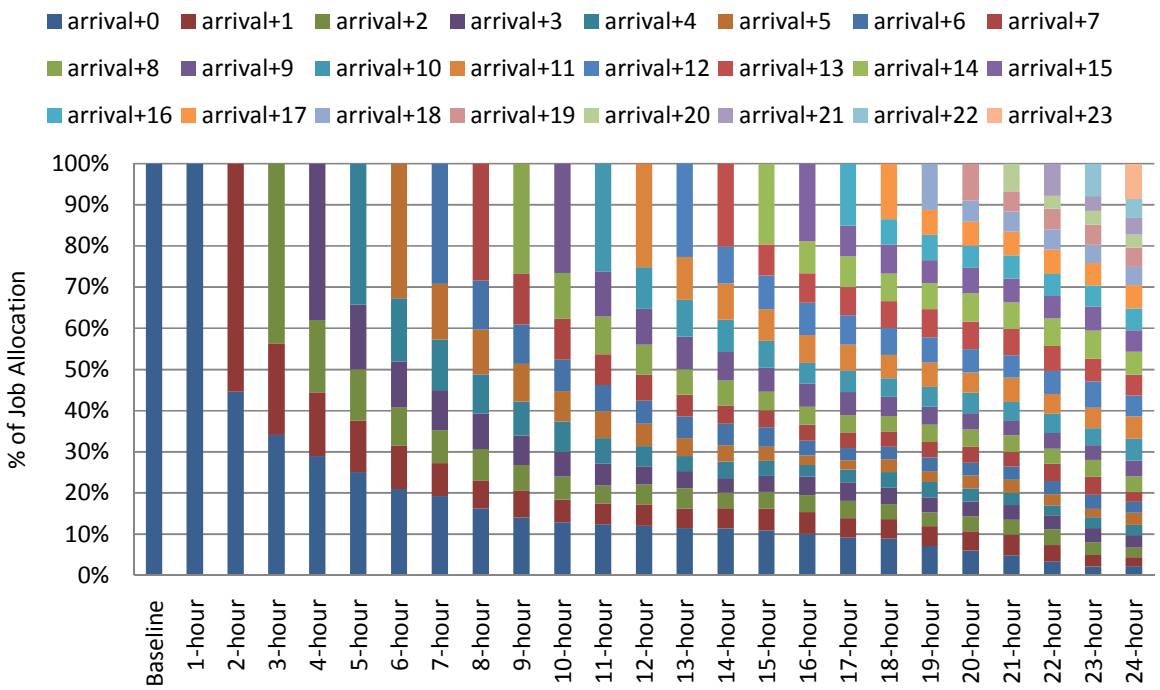


Figure 55: Comparison of Temporal Workload Distribution between Baseline & GBwLA (Deadline 1-24 Hours)

7.3.3 Carbon Price

In our aggregate cost function (see section 4.3.3), we use carbon price as an aggregate coefficient. The value of carbon price influences how much carbon cost contributes to aggregate cost; the higher the price, the more influential the carbon cost.

To determine how sensitive our results are to carbon price, we run simulations for GB and GBwLA with identical input parameters except that the carbon price is set from \$10/tCO_2e up to \$100 /tCO_2e, increased by \$10 after each simulation. We choose GB and GBwLA because they optimize for both electricity cost and carbon cost, allowing us to assess how the aggregation of these two costs respond to changes in carbon price. As for the other heuristics, their outcomes do not depend on carbon price since their aggregate cost functions only include one type of cost (i.e. electricity cost for GEC & GECwLA, & carbon cost for GCF & GCFwLA.

GB

As shown in Figure 56-57, there is a shift in cost structure with savings moving from electricity cost to carbon cost as carbon price increases. Interestingly the two costs move in opposite directions at

different speeds with carbon cost dropping faster than electricity cost rising, suggesting that savings on carbon cost is more sensitive to changes in carbon price.

As carbon price increases from \$1 to \$10 per tCO_2e, we see a small rise in average electricity cost per job but a significant drop in average carbon cost per job. Under GB, the average electricity cost per job rises as much as 3.93% (i.e. 93.75% – 89.82%) of the baseline while the average carbon cost per job drops as much as 21.8% of the baseline.

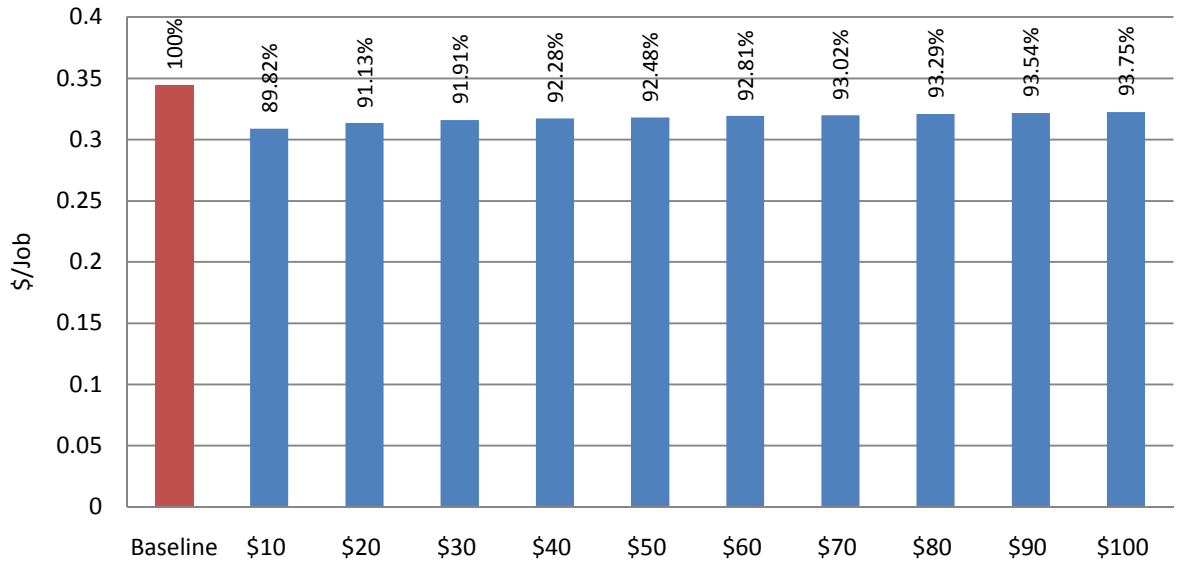


Figure 56: Comparison of Average Electricity Cost per Job between Baseline & GB (Carbon Price \$10-\$100 per tCO_2e)

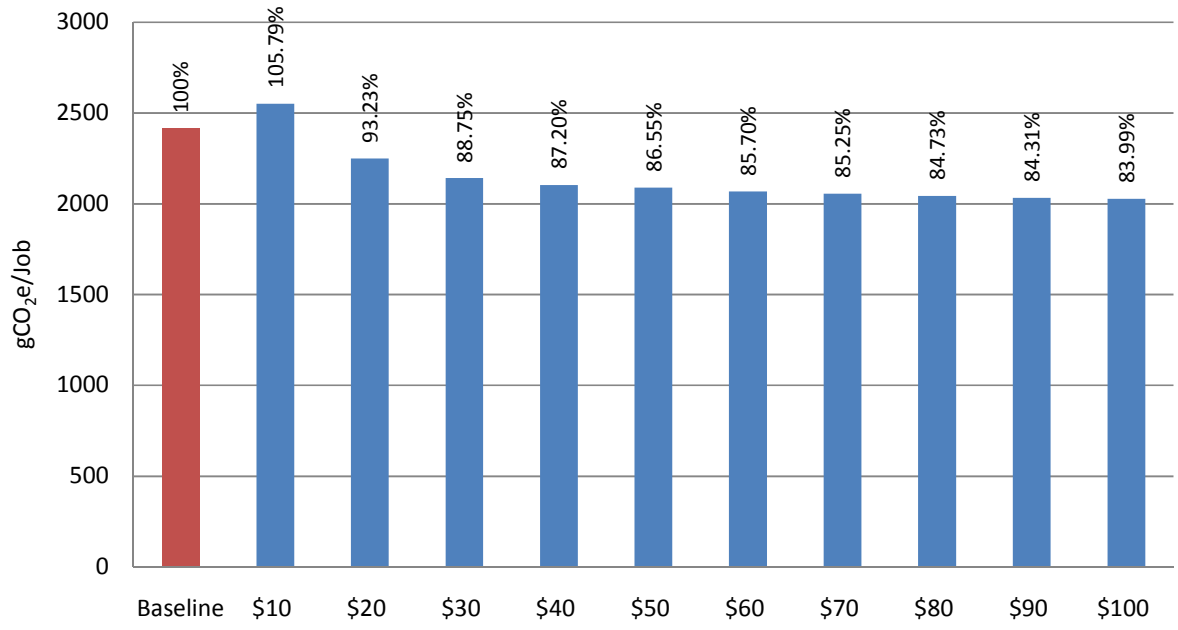


Figure 57: Comparison of Average Carbon Cost per Job between Baseline & GB (Carbon Price $10-$100 per tCO_2e)

To better understand where the savings come from, we examine the geographical and temporal distributions of workloads corresponding to each carbon price.

Figure 58 & 59 show how distributions of workloads in space and time respond to higher carbon price. They complement the previous figures by describing exactly where and when workloads were scheduled. For example, by looking at figure 56 & 57, we know that when the carbon price is set to $30 per tCO_2e, GB reduces 8.09% of electricity cost and 11.25% of carbon cost. But what we do not know is where and when the workload were scheduled. Figure 58 & 59 give us the answer—most of the workloads were scheduled at Ontario, SCE, & PGAE. Also, all of the workloads were scheduled at their immediate hour.

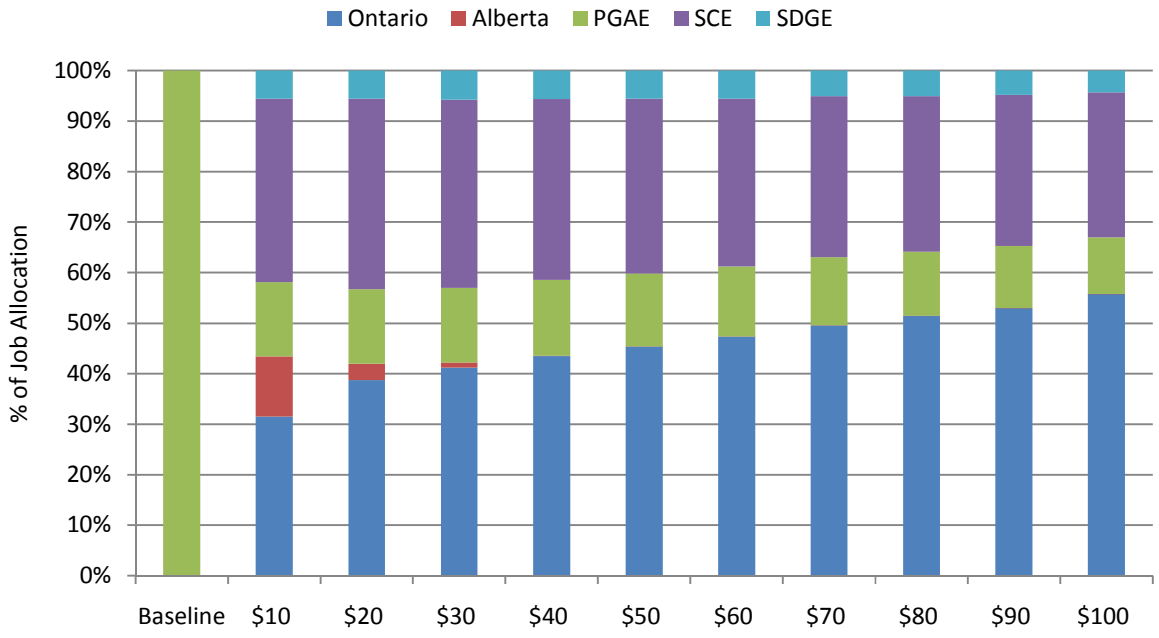


Figure 58: Comparison of Geographical Workload Distribution between Baseline & GB (Carbon Price $10-$100 per tCO_2e)

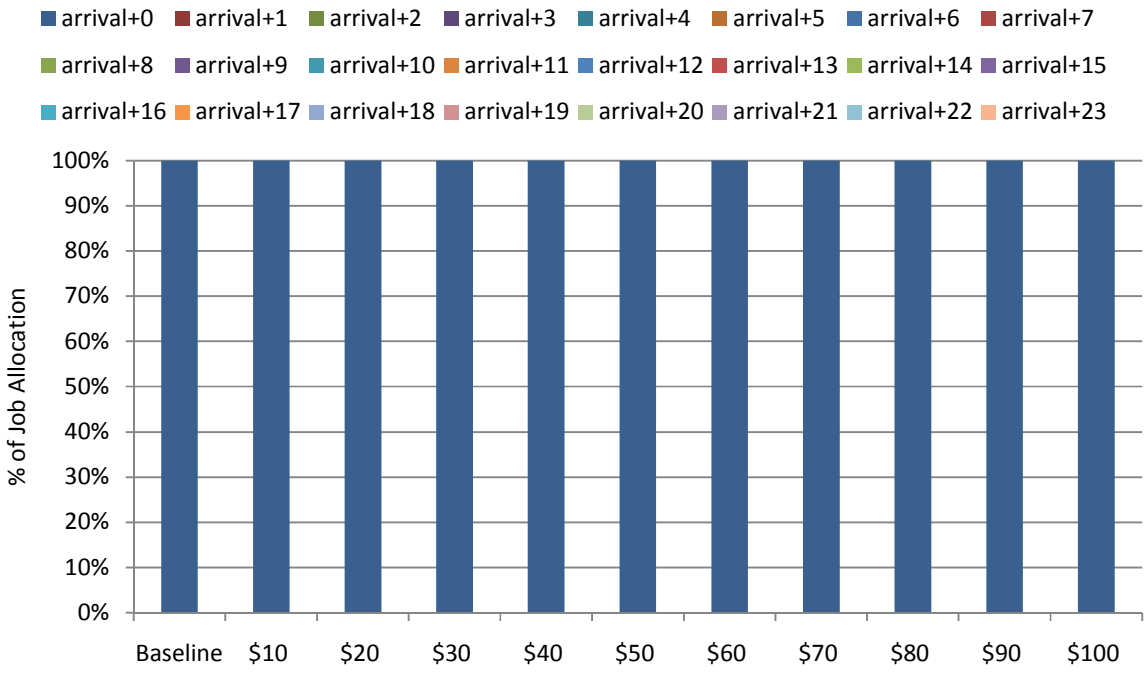


Figure 59: Comparison of Temporal Workload Distribution between Baseline & GB (Carbon Price $10-$100 per tCO2e)

GBwLA

As shown in Figure 60 & 61, there is a shift in cost structure with savings moving from electricity cost to carbon cost as carbon price increases. Interestingly the two costs move in opposite directions at different speeds with carbon cost dropping faster than electricity cost rising, suggesting that savings on carbon cost is more sensitive to changes in carbon price.

As carbon price increases from \$1 to \$10 per tCO_2e, we see a small rise in average electricity cost per job but a significant drop in average carbon cost per job. Under GBwLA, the average electricity cost per job rises as much as 3.98% (i.e. 50.93% - 46.95%) while the average carbon cost per job drops as much as 18.15% under GBwLA.

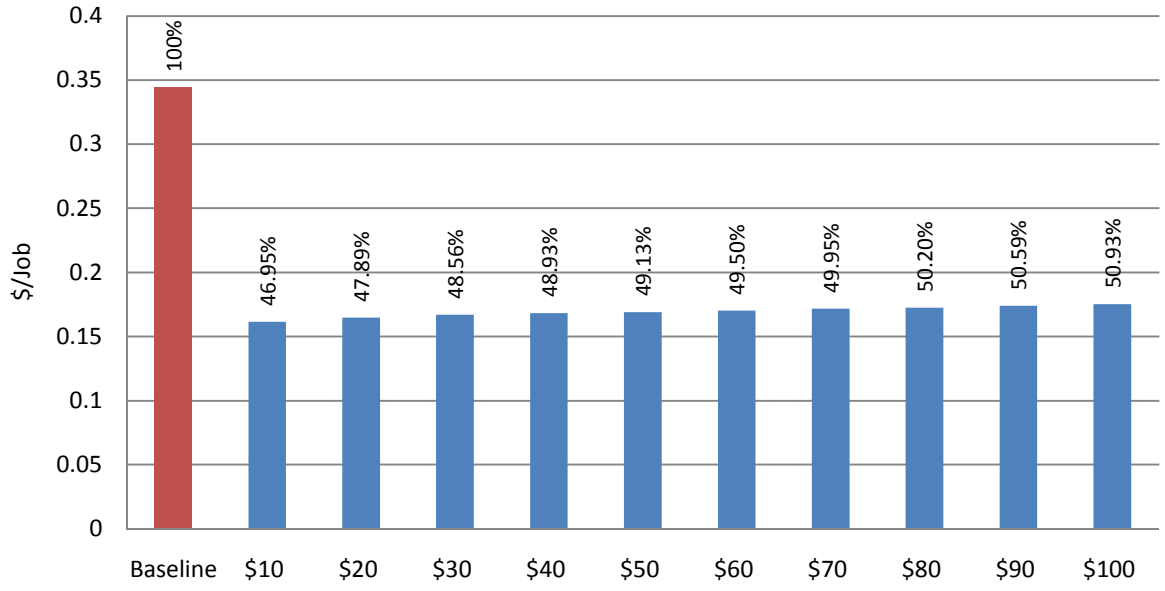


Figure 60: Comparison of Average Electricity Cost per Job between Baseline & GBwLA (Carbon Price \$10-\$100 per tCO_2e)

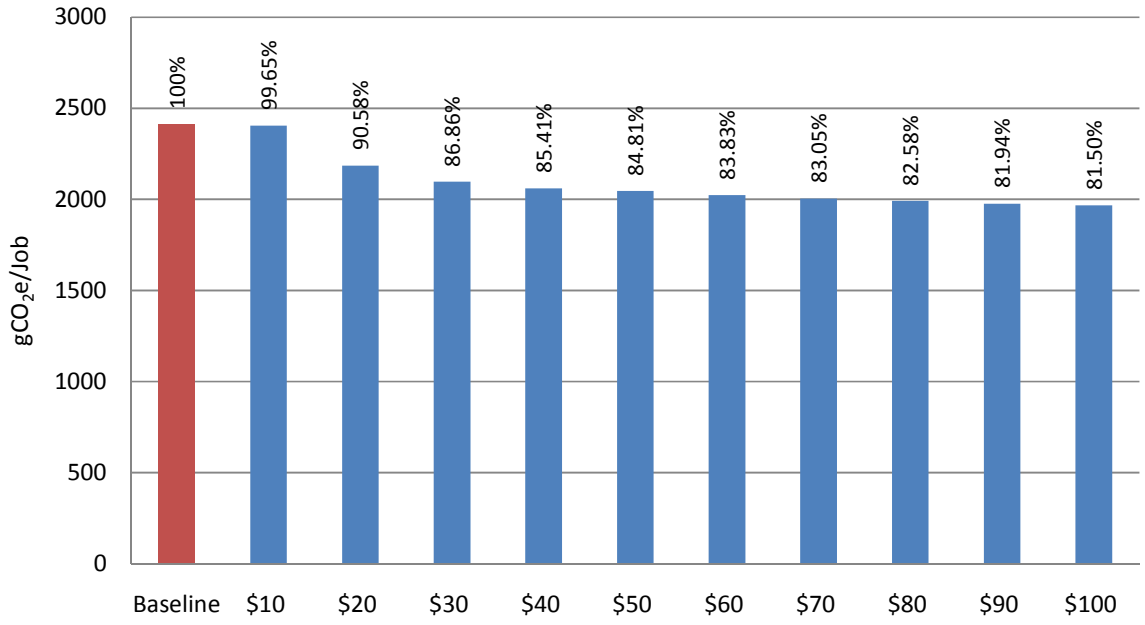


Figure 61: Comparison of Average Carbon Cost per Job between Baseline & GBwLA (Carbon Price $10-$100 per tCO_2e)

To better understand where the savings come from, we examine the geographical and temporal distributions of workloads corresponding to each carbon price.

Figure 62 & 63 show how distributions of workloads in space and time respond to higher carbon price. They complement the previous figures by describing exactly where and when workloads were scheduled. For example, by looking at figure 60 & 61, we know that when the carbon price is set to $30 per tCO_2e, GBwLA reduces 51.07% of the electricity cost and 13.14% of the carbon cost. But what we do not know is where and when the workload were scheduled. Figure 62 & 63 give us the answer—most of the workloads were scheduled at Ontario, SCE, & PGAE. Also, most of the workloads were scheduled at their immediate hour.

While more jobs are routed to locations with cleaner generation mixes such as Ontario and PGAE as carbon price increases, the temporal distribution of workloads remains largely the same.

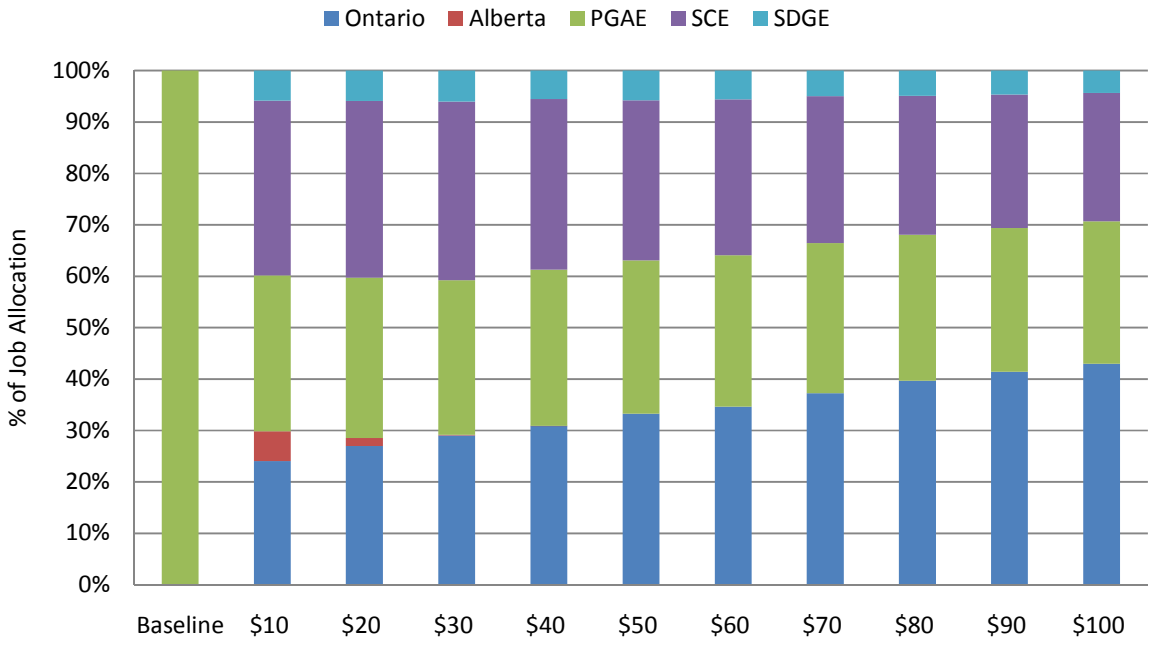


Figure 62: Comparison of Geographical Workload Distribution between Baseline & GBwLA (Carbon Price $10-$100 per tCO_2e)

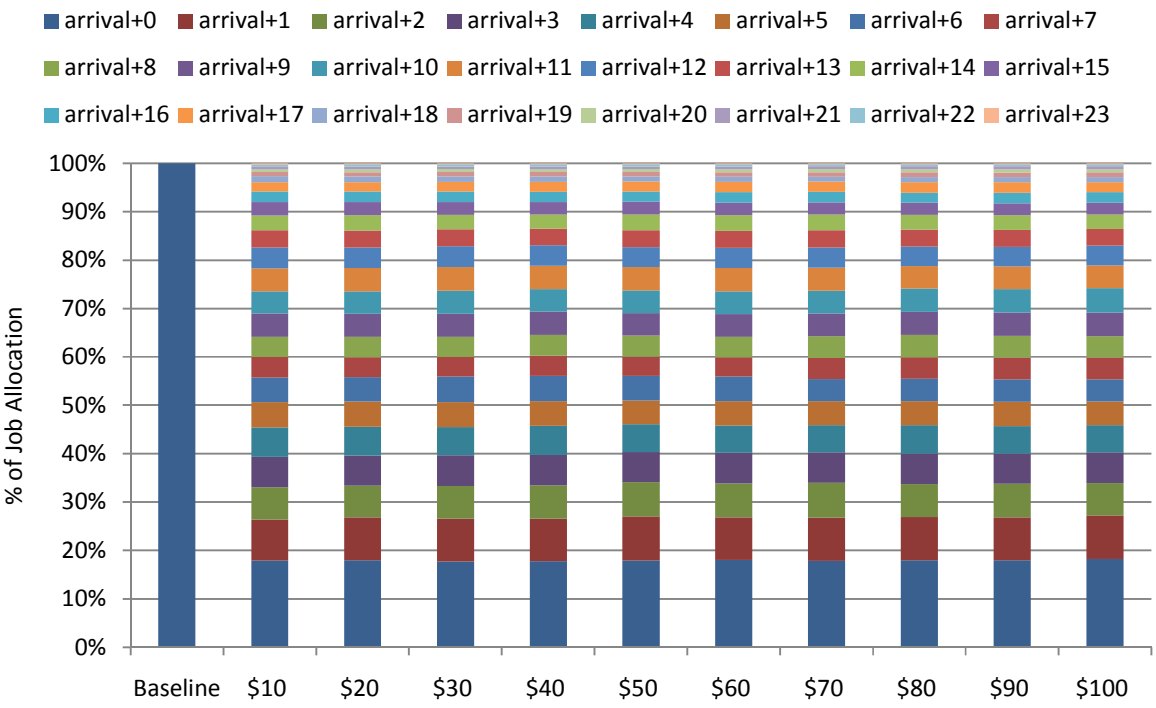


Figure 63: Comparison of Temporal Workload Distribution between Baseline & GBwLA (Carbon Price $10-$100 per tCO2e)

7.3.4 Findings & Implications

Our optimization heuristics provide different trade-off options between electricity cost and carbon cost. Cloud providers can use GEC/GECwLA to minimize electricity cost, or GCF/GCFwLA to minimize carbon cost, or GB/GBwLA to reduce both costs. Being able to support multiple trade-off options allow Cloud providers to cater to Cloud users' needs.

Optimizing solely for electricity cost leads to substantial penalty on carbon cost and vice versa. The difference between the highest and lowest costs is as much as 41.5% for electricity cost and 42.25% for carbon cost when look-ahead is used. Cloud providers can reduce both costs by choosing a more balanced workload distribution strategy such as GB and GBwLA.

Compared to the baseline, our heuristics produces substantial cost savings. By allocating jobs geographically our heuristics can save up to 10.64% on electricity cost, or 21.25% on carbon cost, or 8.09% on electricity cost and 11.25% on carbon cost simultaneously. Introducing job delay in addition to geographical allocation results in even higher savings—up to 53.35% on electricity cost, or 29.13% on carbon cost, or 51.44% on electricity cost and 13.14% on carbon cost simultaneously. The shifting of jobs with flexible deadlines in space and time implies that Cloud providers may need to adjust their back-up and maintenance strategies to accommodate new flows of workloads arriving at locations and times that have not occurred before.

Increasing job deadline results in more savings. Compared to carbon cost, electricity cost is more sensitive to changes in job deadline; it drops at a faster pace when job deadline gets longer. Cloud provider can perform cost-benefit analysis to determine whether additional savings is worth the delay.

High carbon prices push workloads towards data center locations with lower emission intensities but have little effect on the timing of their execution. Thus, in a business environment where carbon is substantially priced, it is financially and environmentally beneficial for Cloud providers to locate their data centers in regions powered by clean generation sources.

Overall, the results from our simulations have shown quantitatively the extent of dollar and carbon savings that can be achieved with our optimization heuristics. They have also demonstrated how data center location and timing of a workload interact with each other.

8 Conclusion

8.1 Summary of Contributions

Cloud Computing provides a novel way for businesses to procure their IT needs. Its elastic scalability and on-demand provisioning enables a shift from capital expenditures to operating expenses, giving businesses the technological agility they need to respond to an ever-changing marketplace. The rapid adoption of Cloud Computing, however, poses a unique challenge to Cloud providers—their already very large electricity bill and carbon footprint will get larger as they expand; managing both costs is therefore essential to their growth.

In this thesis, we endeavor to address the above challenge. Recognizing the existence of Cloud data centers in multiple locations and the differences in electricity price and emission intensity among these locations over time, we propose an optimization framework that couples workload distribution with signals on electricity price and emission intensity for financial and environmental benefits. The framework is comprised of three key components—an optimization model, an aggregate cost function, and six workload scheduling heuristics. The model provides a way to gain insights into how cost savings can be achieved by coupling workload distribution with time-varying signals on electricity price and emission intensity. The aggregate cost function combines different types of costs such as electricity cost and carbon cost into one figure of merit, enabling optimization for multiple costs simultaneously. And the heuristics provide different scheduling strategies that Cloud providers can implement to achieve different trade-offs.

Built on previous work, the thesis extends the literature in two ways. Firstly, it addresses the trade-off problem between electricity cost and carbon cost when routing workloads to geographically distributed data centers, enabling Cloud providers to reduce both costs simultaneously. In comparison, previous work focuses on either electricity cost or carbon cost, but not both. Secondly, it studies the shifting of workloads in both time and space, uncovering new opportunities for Cloud providers to cut costs. Previous work only explores shifting workloads in space.

To evaluate cost savings, we run simulations with 5 data centers located across North America over a period of 81 days. We use historical data on electricity price, emission intensity, and workload collected from market operators and research data archives. We find that our framework can produce substantial cost savings, especially when workloads are distributed both geographically and temporally—up to

53.35% on electricity cost, or 29.13% on carbon cost, or 51.44% on electricity cost and 13.14% on carbon cost simultaneously.

8.2 Limitations & Future Work

To reduce complexity the thesis omits or simplifies certain aspects of workload distribution across geographically distributed data centers. Addressing these shortcomings to make this work more applicable in practice is an important direction of our future research.

Network Related Costs. To take advantage of cheaper and/or cleaner electricity, our algorithms may need to route workloads to distant data center locations. The increase in network latency and especially bandwidth cost needs to be taken into account in addition to the savings on electricity and carbon costs. A more comprehensive model therefore needs to incorporate network related costs in the optimization process. Network latency can be modeled as a function of client-server distance and the number of routers and switches through which the traffic needs to pass. Bandwidth cost can be modeled as a function of the network traffic volume at a given location. More discussion on this topic can be found in (Qureshi et al., 2009), (Rao et al., 2010), and (Kien Le et al., 2010).

Data Center Efficiency. Our optimization model focuses on cost factors that are external to Cloud data centers (i.e. electricity price and emission intensity). It is clear, however, that internal factors such as the efficiency of data center equipment play an equally important role in constituting the cost of executing a workload. A more detailed model needs to take into account server heterogeneity, time-varying non-IT load, and energy efficiency of all server components, not just CPU. More discussion on this topic can be found in (Sankaranarayanan et al., 2011) and (Garg et al., 2011).

Cost Aggregation. While our weighted sum function allows decision makers to favor a criterion (i.e. cost component) by giving it a heavier weight, it does not prevent the favored criterion from being traded off with other criteria. For example, consider a scenario where *network latency* is of paramount importance to a Cloud provider and they are not willing to tradeoff satisfaction of *electricity and carbon costs* until perhaps they attain some level of satisfaction of *network latency*. More advanced aggregation operators such as Prioritized Weighted Aggregation or PWA operator (Yan, Huynh, Nakamori, & Murai, 2011) can satisfy such requirements. PWA allows decision makers to prioritize criteria, ensuring that higher-priority criteria are satisfied first. Another unique advantage of PWA operator is that it can capture various interrelationships between criteria (i.e. cost components), ranging from “all the criteria must be satisfied” to “at least one of the criteria must be satisfied.”

PWA operator is based on the ordered weighted averaging (OWA) operator (Yager, 1988) and triangular norms (t-norms). Given a set of criteria partitioned into distinct priority levels, t-norms are used to induce a priority weight for each level while the OWA operator is used to aggregate the criteria in the same level. The final aggregated value is obtained by summing up the product of the priority weight and aggregated value of all priority levels.

Uncertainty. Our optimization model assumes perfect knowledge of external cost factors such electricity price & emission intensity as well as internal cost factors such energy usage of data center equipment. In practice, however, input to the model is likely imperfect and involves some level of uncertainty. Fuzzy logic can be an effective tool for reasoning under uncertainty.

Fuzzy logic is based on the fuzzy set theory developed in 1965 by Lotfi Zadeh. It can be considered as a superset of conventional logic which allows for degrees of truth—truth values between true and false. Fuzzy logic is an instance of soft computing whose goal is to mimic the ability of the human mind to effectively employ modes of reasoning that are approximate rather than exact (Zadeh, 1994). It allows the mapping of different criteria values into linguistic values characterizing levels of satisfaction. A linguistic variable is a variable whose values are words or sentences in a natural or synthetic language. Linguistic variables are used to formulate fuzzy rules in the form of "IF-THEN" whose antecedents and consequents are fuzzy-logic statements that represent the knowledge or control strategies of a fuzzy rule-based system.

The above list is by no means exhaustive. Some other simplifying assumptions, such as the impact of workload shifting on local power grids and the scalability of workloads, are described in section 3.5 and deserve further investigation.

9 Appendix A – Historical Data on Workload, Electricity Price, & Emission Intensity

Workload Trace

WORKLOAD_TRACE.zip

Electricity Price

ELECTRICITY_PRICES_EST.zip

Emission Intensity

EMISSION_INTENSITIES_EST.zip

10 Appendix B – Simulation Results


Results.zip

11 References

AESO. (2011a). *Actual / forecast.* Retrieved 12/14, 2011, from http://ets.aeso.ca/ets_web/ip/Market/Reports/ActualForecastWMRQHReportServlet

AESO. (2011b). *Alberta energy: Electricity.* Retrieved 11/15, 2011, from http://www.energy.alberta.ca/OurBusiness/electricity.asp

AESO. (2011c). *Market & system reporting.* Retrieved 11/18, 2011, from http://ets.aeso.ca

AESO. (2011d). *Wind power / AIL data.* Retrieved 11/15, 2011, from http://www.aeso.ca/gridoperations/20544.html

Andersson, J. (2000). *A survey of multiobjective optimization in engineering design.* (No. LiTH-IKP-R-1097).Linköping University. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.8.5638&rep=rep1&type=pdf

Armbrust, M., Fox, A., Rean, G., Joseph, A. D., Katz, R., Konwinski, A., . . . Zaharia, M. (2009). *Above the clouds: A berkeley view of cloud computing* EECS Department, University of California, Berkeley. Retrieved from http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html

ASHRAE TC 9.9. (2011). *2011 thermal guidelines for data processing environments – expanded data center classes and usage guidance.* ().ASHRAE. Retrieved from http://tc99.ashraetcs.org/documents/ASHRAE%20Whitepaper%20-%202011%20Thermal%20Guidelines%20for%20Data%20Processing%20Environments.pdf

Barroso, L. A., & Holzle, U. (2007). The case for energy-proportional computing. *Computer, 40*(12), 33-37.

Beloglazov, A., Buyya, R., Lee, Y. C., & Zomaya, A. (2011). A taxonomy and survey of energy-efficient data centers and cloud computing systems. *ADVANCES IN COMPUTERS, 82*, 47. doi:10.1016/B978-0-12-385512-1.00003-7

Berl, A., Gelenbe, E., Di Girolamo, M., Giuliani, G., De Meer, H., Dang, M. Q., & Pentikousis, K. (2010). Energy-efficient cloud computing. *The Computer Journal, 53*(7), 1045-1051. doi:10.1093/comjnl/bxp080

Borshchev, A., & Filippov, A. (2004). From system dynamics and discrete event to practical agent based modeling: Reasons, techniques, tools. Switzerland. *, The 22nd International Conference of the System Dynamics Society*

Buchbinder, N., Jain, N., & Menache, I. (2011). *Online job-migration for reducing the electricity bill in the cloud.* (). Retrieved from http://research.microsoft.com/en-us/um/people/navendu/infocom2011-tr.pdf

CAISO. (2011a). *Daily renewables watch.* Retrieved 15/11, 2011, from http://www.caiso.com/market/Pages/ReportsBulletins/DailyRenewablesWatch.aspx

CAISO. (2011b). *Market issues and performance annual report.* ().CAISO. Retrieved from http://www.caiso.com/Documents/2010AnnualReportonMarketIssuesandPerformance.pdf

CAISO. (2011c). *Reports and bulletins archive.* Retrieved 11/18, 2011, from http://www.caiso.com/market/Pages/ReportsBulletins/ReportsBulletinsArchive/Default.aspx

CAISO. (2011d). *Today's outlook.* Retrieved 12/12, 2011, from http://www.caiso.com/Pages/TodaysOutlook.aspx

CEA. (2006). *Power generation in canada.* ().CEA. Retrieved from http://www.electricity.ca/media/pdfs/backgrounders/HandBook.pdf

CEA. (2010). *Industry data.* Retrieved 11/15, 2011, from http://www.electricity.ca/industry-issues/electricity-in-canada/industry-data.php

Chabarek, J., Sommers, J., Barford, P., Estan, C., Tsiang, D., & Wright, S. (2008). Power awareness in network design and routing. Paper presented at the *INFOCOM 2008. the 27th Conference on Computer Communications. IEEE,* 457-465.

Dean, J., & Ghemawat, S. (2008). MapReduce: Simplified data processing on large clusters. *Commun.ACM, 51*(1), 107-113. doi:http://doi.acm.org/10.1145/1327452.1327492

EIA. (2009). *Energy in brief.* Retrieved 11/15, 2011, from http://www.eia.gov/energy_in_brief/renewable_energy.cfm

EPA. (2007). *Report to congress on server and data center energy efficiency.* ().EPA. Retrieved from http://www.energystar.gov/index.cfm?c=prod_development.server_efficiency_study;

Farhat, A. A. M., & Ugursal, V. I. (2010). Greenhouse gas emission intensity factors for marginal electricity generation in canada. *International Journal of Energy Research, 34*(15), 1309-1327. doi:10.1002/er.1676

Feitelson, D. (2008). *The LLNL thunder log, parallel workloads archive.* Retrieved 07/15, 2011, from http://www.cs.huji.ac.il/labs/parallel/workload/l_llnl_thunder/index.html

Garg, S. K., Yeo, C. S., Anandasivam, A., & Buyya, R. (2011). Environment-conscious scheduling of HPC applications on distributed cloud-oriented data centers. *Journal of Parallel and Distributed Computing, 71*(6), 732. doi:DOI: 10.1016/j.jpdc.2010.04.004"

Gilli, M., & Winker, P. (2009). Heuristic optimization methods in econometrics., 81-119. doi:10.1002/9780470748916.ch3

Grabisch, M. (1996). The application of fuzzy integrals in multicriteria decision making. *European Journal of Operational Research, 89*(3), 445-456. Retrieved from http://ideas.repec.org/a/eee/ejores/v89y1996i3p445-456.html

Green Grid. (2007). *GREEN GRID METRICS: Describing datacenter power efficiency.* ().Green Grid. Retrieved from http://www.thegreengrid.org/~/media/WhitePapers/Green_Grid_Metrics_WP.ashx?lang=en

Hamoud, G., & Bradley, I. (2004). Assessment of transmission congestion cost and locational marginal pricing in a competitive electricity market. *Power Systems, IEEE Transactions on, 19*(2), 769-775.

Harms, R., & Yamartino, M. (2010). *The economics of the cloud.* (White Paper).Microsoft. Retrieved from http://www.microsoft.com/presspass/presskits/cloud/docs/The-Economics-of-the-Cloud.pdf

Hoelzle, U., & Barroso, L. A. (2009). The datacenter as a computer: An introduction to the design of warehouse-scale machines. In (1st ed.,) Morgan and Claypool Publishers.

IBM. (2011). *IBM global CIO study.* ().IBM. Retrieved from http://www-935.ibm.com/services/c-suite/cio/study.html

IDC. (2010). *IDC cloud research.* Retrieved 5/25, 2011, from http://www.idc.com/prodserv/idc_cloud.jsp

IESO. (2011a). *Market data.* Retrieved 11/18, 2011, from http://ieso.com/imoweb/marketdata/marketData.asp

IESO. (2011b). *Ontario demand and market prices.* Retrieved 12/10, 2011, from http://www.ieso.ca/imoweb/siteShared/demand_price.asp?sid=ic

IESO. (2011c). *Supply overview.* Retrieved 11/15, 2011, from http://www.ieso.ca/imoweb/media/md_supply.asp

IRC. (2009). *2009 state of the markets report.* ().IRC. Retrieved from http://www.isorto.org/atf/cf/%7B5B4E85C6-7EAC-40A0-8DC3-003829518EBD%7D/2009%20IRC%20State%20of%20Markets%20Report.pdf

ITU. (2011). *Free statistics.* Retrieved 05/07, 2011, from http://www.itu.int/ITU-D/ict/statistics/

Kien Le, Bianchini, R., Nguyen, T. D., Bilgir, O., & Martonosi, M. (2010). Capping the brown energy consumption of internet services at low cost. Paper presented at the *Green Computing Conference, 2010 International,* 3-14.

Koomey, J., Belady, C., Patterson, M., & Santos, A. (2009). *Assessing trends over time in performance, costs, and energy use for servers.* ().

Liu, L., Wang, H., Liu, X., Jin, X., He, W. B., Wang, Q. B., & Chen, Y. (2009). GreenCloud: A new architecture for green data center. Paper presented at the *Proceedings of the 6th International Conference Industry Session on Autonomic Computing and Communications Industry Session,* Barcelona, Spain. 29-38. doi:http://doi.acm.org.proxy.bib.uottawa.ca/10.1145/1555312.1555319

Marler, R. T., & Arora, J. S. (2004). *Survey of multi-objective optimization methods for engineering* Springer Berlin / Heidelberg. doi:10.1007/s00158-003-0368-6

McCullough, J. C., Agarwal, Y., Chandrashekar, J., Kuppuswamy, S., Snoeren, A. C., & Gupta, R. K. (2011). Evaluating the effectiveness of model-based power characterization. Paper presented at the *Proceedings of the 2011 USENIX Conference on USENIX Annual Technical Conference,* Portland, OR. 12-12. Retrieved from http://dl.acm.org/citation.cfm?id=2002181.2002193

McKinsey. (2008). *Revolutionizing data center efficiency.* ().McKinsey & Company. Retrieved from http://www.mckinsey.com/clientservice/bto/pointofview/Revolutionizing.asp

Meisner, D., Gold, B. T., & Wenisch, T. F. (2009). PowerNap: Eliminating server idle power. *SIGPLAN Not., 44*(3), 205-216. doi:http://doi.acm.org/10.1145/1508284.1508269

Mell, P., & Grance, T. (2010). *The NIST definition of cloud computing.* ().NIST. Retrieved from http://csrc.nist.gov.proxy.bib.uottawa.ca/groups/SNS/cloud-computing/

Mellah, H., & Sanso, B. (2009). Review of facts, data and proposals for a greener internet. Paper presented at the *Broadband Communications, Networks, and Systems, 2009. BROADNETS 2009. Sixth International Conference on,* 1-5.

NERC. (2011). *Regional entities.* Retrieved 11/15, 2011, from http://www.nerc.com/page.php?cid=1|9|119

Parkhill, D. (1966). *The challenge of the computer utility* Addison-Wesley Educational Publishers Inc.

Pelley, S., Meisner, D., Wenisch, T., & VanGilder, J. (2009). *Understanding and abstracting total data center power.* (). Retrieved from http://www.eecs.umich.edu/~spelley/Publications/weed09.pdf

Qureshi, A. (2010). *Power-demand routing in massive geo-distributed systems.* Massachusetts Institute of Technology). Retrieved from http://people.csail.mit.edu/asfand/papers/aqureshi_phd_csail_2010.pdf

Qureshi, A., Weber, R., Balakrishnan, H., Guttag, J., & Maggs, B. (2009). Cutting the electric bill for internet-scale systems. *SIGCOMM Comput.Commun.Rev., 39*(4), 123-134. doi:http://doi.acm.org/10.1145/1594977.1592584

Rao, L., Liu, X., Xie, L., & Liu, W. (2010). Minimizing electricity cost: Optimization of distributed internet data centers in a multi-electricity-market environment. Paper presented at the *INFOCOM, 2010 Proceedings IEEE,* 1-9.

Sankaranarayanan, A. N., Sharangi, S., & Fedorova, A. (2011). Global cost diversity aware dispatch algorithm for heterogeneous data centers. Paper presented at the *Proceeding of the Second Joint WOSP/SIPEW International Conference on Performance Engineering,* Karlsruhe, Germany. 289-294. doi:http://doi.acm.org/10.1145/1958746.1958787

Scaramella, J., & Eastwood, M. (2007). *Solutions for the datacenter's thermal challenges.* ().IDC.

Sovacool, B. K. (2008). Valuing the greenhouse gas emissions from nuclear power: A critical survey. *Energy Policy, 36*(8), 2950. doi:DOI: 10.1016/j.enpol.2008.04.017"

The Climate Group. (2008). *Smart 2020.* ().The Climate Group. Retrieved from http://www.smart2020.org/publications

Varsamopoulos, G., Abbasi, Z., & Gupta, S. K. S. (2010). Trends and effects of energy proportionality on server provisioning in data centers. Paper presented at the *High Performance Computing (HiPC), 2010 International Conference on,* 1-11.

Verma, A., Ahuja, P., & Neogi, A. (2008). In Issarny V., Schantz R.(Eds.), *pMapper: Power and migration cost aware application placement in virtualized systems* Springer Berlin / Heidelberg. doi:10.1007/978-3-540-89856-6_13

Yager, R. R. (1988). On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Trans.Syst.Man Cybern., 18*(1), 183-190. doi:10.1109/21.87068

Yan, H., Huynh, V., Nakamori, Y., & Murai, T. (2011). On prioritized weighted aggregation in multi-criteria decision making. *Expert Systems with Applications, 38*(1), 812-823. doi:10.1016/j.eswa.2010.07.039

Zadeh, L. A. (1994). Fuzzy logic, neural networks, and soft computing. *Commun.ACM, 37*(3), 77-84. doi:http://doi.acm.org/10.1145/175247.175255